# Generalized Language Identification

Marco Hoyin Lui

Submitted in total fulfilment of
the requirements of the degree of
Doctor of Philosophy

July 2014

Department of Computing and Information Systems

The University of Melbourne

Produced on Archival Quality Paper

Thesis advisor(s)

**Timothy Baldwin**

Author

**Marco Hoyin Lui**

**Generalized Language Identification**

# Abstract

Language identification is the task of determining the natural language that a document or part thereof is written in. The central theme of this thesis is *generalized* language identification, and deals with eliminating the assumptions that limit the applicability of language identification techniques to specific settings that may not be representative of real-world use cases for automatic language identification techniques. Research to date has treated language identification as a supervised machine learning problem, and in this thesis I argue that such a characterization is inadequate, showing how standard document representations do not take into account the variation in a language between different sources of text, and developing a representation that is robust to such variation. I also develop a method that allows for language identification of multilingual documents, i.e. documents that contain text in more than one language. Finally, I investigate the robustness of existing off-the-shelf language identification methods on a novel and challenging domain.

# Declaration

This is to certify that:

(i) the thesis comprises only my original work towards the PhD except where indicated in the Preface,

(ii) due acknowledgement has been made in the text to all other material used,

(iii) the thesis is fewer than 100 000 words in length, exclusive of tables, maps, bibliographies and appendices.

# Citations to Previously Published Work

Chapter 4 and Chapter 5 are based on work previously published as:

Lui, Marco, and Timothy Baldwin. 2011. Cross-domain Feature Selection for Language Identification. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, 553 – 561, Chiang Mai, Thailand.

Chapter 6 is based on work previously published as:

Lui, Marco, Jey Han Lau, and Timothy Baldwin. 2014. Automatic Detection and Language Identification of Multilingual Documents. *Transactions of the Association for Computational Linguistics*, 2(Feb):27 – 40.

Chapter 7 is based on work previously published as:

Lui, Marco, and Timothy Baldwin. 2014. Accurate Language Identification of Twitter Messages. In *Proceedings of the 5th Workshop on Language Analysis in Social Media*, 17 – 25, Gothenburg, Sweden.

# Acknowledgments

This thesis was a substantial undertaking, and took a little longer and came out a little thicker than I would have liked. It would not have been possible without the support and understanding of those close to me. First and foremost I have to thank my wife Nicole, who celebrated successes with me and and tolerated me through failures and disappointments. I also want to thank my parents for giving me the possibity to become who I am today. I most definitely need to thank my supervisor Tim, for mentoring me from a naive undergraduate through to being a more complete and hopefully marginally wiser person. I wish to thank Justin, Steven and Karin, who served on my thesis committee and whose insights and suggestions have helped shape my thesis and my research over the past four years. I'm also grateful to my anonymous and slightly-less anonymous examiners for their positive reviews and feedback that I have incorporated into the final version of this manuscript. Thanks also to Paul Cook, David Martinez and Diana McCarthy, from whom I learned so much by working with. I'm also grateful to the people around the world that have hosted me and made my research experience richer and more fulfilling: Chin-Yew Lin's group at MSRA, Hwee Tou Ng and Min-Yen Kan at NUS, Francis Bond at NTU, Marcos Zampieri at Saarland University, Jörg Tiedemann and Joakim Nivre at Uppsala University. I want to thank NICTA, UniMelb and MSRA, whose financial support made it possible for me to travel and engage researchers domestically and internationally, allowing me to visit Sydney, Brisbane, Kioloa, Ettalong Beach, Dunedin, Beijing, Chiang Mai, Singapore, Jeju Island, Saarland, Uppsala and Gothenburg. My life has certainly been made richer by the people I met and the experiences I had. Last but not least, I would like to thank friends and colleagues from CSSE (now CIS), 6.05 and 8.19,

from whom I have also learned a great deal. In alphabetical order: Andy, Bahar, Bo, Florian, Goce, Graeme, Jey Han, Jim, Karl, Li, Long, Mel, Ned, Oliver, Rebecca, Richard, Sergey, Simone, Spandana, Sumukh, Willy. I hope, despite our time as students coming to a close, that the breakfasts and barbeques may still continue.

Marco Lui, February 2015

*Dedicated to my father, Lui Kin Yip (1951-2013).*

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Language identification (LangID) is the task of determining the natural language that a document or part thereof is written in. The problem of LangID is one that is intuitively familiar, since one of the characteristics of being human is the ability to communicate complex and sophisticated thoughts and ideas, and this is only possible through the use of a common language. People are generally quickly able to recognize languages that they are familiar with. Table 1.1 presents excerpts from Wikipedia articles in different languages on the topic of natural language processing, labeled according to the language they are written in. Without referring to the labels, readers of this thesis will certainly have recognized at least one language in Table 1.1, and many are likely to be able to identify all the languages therein.

Research into LangID aims to mimic this human ability to recognize specific languages. Over the years, a number of computational approaches have been developed that, through the use of specially-designed algorithms and data structures, are able to infer the language being used without the need for human intervention. In a way,

| | |
|---|---|
| English | Natural language processing is a field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages. |
| Italian | L'Elaborazione del linguaggio naturale è il processo di trattamento automatico mediante un calcolatore elettronico delle informazioni scritte o parlate nel linguaggio umano o naturale. |
| Chinese | **自然語言處理是人工智慧和語言學領域的分支學科。** |
| Japanese | 自然言語処理は、人間が日常的に使っている自然言語をコンピュータに処理させる一連の技術であり、人工知能と言語学の一分野である。 |

Table 1.1: Excerpts from Wikipedia articles on natural language processing in different languages.

the capability of such systems is super-human. An average person may be able to identify a handful of languages, and a trained linguist or translator may be familiar with dozens, but most of us will have experienced at some point an encounter with a language that is alien to us. However, LangID research aims to develop systems that are able to recognize any human language, a set which numbers in the thousands (Lewis *et al.* 2014).

In a broad sense, LangID applies to any modality of language, including speech and handwritten text, and is relevant for all means of information storage that involve language, digital or otherwise. However, in this thesis, we limit the scope of our investigation to LangID of *documents*, i.e. written text. Furthermore, we assume that this text is stored in a digitally-encoded form, though we do not assume that the exact encoding is known in advance.

Research to date on LangID has generally focused on *monolingual* documents (Hughes *et al.* 2006). In monolingual LangID, the task is to assign each document a

unique language. Some work has reported near-perfect accuracy for LangID of large documents in a small number of languages (Cavnar and Trenkle 1994; Dunning 1994; Grefenstette 1995; Prager 1999a; Teahan 2000), prompting some researchers to label it a "solved task" (McNamee 2005). However, in order to attain such accuracy, simplifying assumptions have to be made, such as the aforementioned monolinguality of each document, as well as assumptions about the type and quantity of data and the number of languages considered. These assumptions do not correspond to the challenges presented by dealing with real-world data. Increased availability of large quantities of textual data from a diverse variety of sources has led to a demand for methods to identify language in settings which diverge greatly from those that have been examined in the literature. The central theme of this thesis, *generalized* language identification, deals with eliminating the assumptions that limit the applicability of LangID methods to specific settings. Our aim is to develop LangID techniques that are effective in a more general context, and so we focus specifically on gaps that exist between the capabilities of existing methods and the difficulties presented by real-world data. We assemble a large collection of language-labeled data from existing corpora for the purposes of this research, and use this data to facilitate the analysis and synthesis of the concepts in LangID research to date, identifying core ideas and common themes, and draw on research from areas such as machine learning, topic modeling and multi-label classification in order to address some of the open issues in LangID.

The ability to accurately detect the language that a document is written in is an enabling technology that increases accessibility of data and has a wide variety

of applications. In natural language processing (NLP), most techniques presuppose that the language of input text is known, and many techniques further assume that all documents are in the same language. In order to apply NLP techniques to real-world data, LangID is typically the first step in order to ensure that only documents in relevant languages are subjected to further processing. Similarly, in information storage and retrieval, it is common to index documents in a multilingual collection by the language that they are written in, and LangID is necessary for document collections where the languages of documents is not known a-priori, such as in data crawled from the World Wide Web. Another application of LangID that predates computational methods is the detection of the language of a document for routing to a suitable translator; this application has become even more prominent due to the advent of machine translation methods. In order for machine translation to be applied to translate a document to a target language, it is generally necessary to know the language being translated from, and this is the task of LangID. LangID also plays a part in helping to bridge an increasing "digital divide" by providing support for the documentation and use of lower-density languages. One area where LangID is frequently used in this regard is in linguistic corpus creation, where LangID is used to process targeted web crawls to collect text resources for lower-density languages (Scannell 2007; Xia *et al.* 2010a; Yamaguchi and Tanaka-Ishii 2012; King and Abney 2013).

In this chapter, we provide a brief introduction to the problem of LangID, discussing the nature of the task as well as the unique challenges presented (Section 1.1). We then discuss some of the open issues in LangID that we tackle as part of our ef-

forts to achieve *generalized* language identification (Section 1.2), and we conclude this chapter with an overview of the structure and content of the remainder of this thesis (Section 1.3).

## 1.1 LangID as Text Categorization

LangID is in some ways a special case of text categorization. Sebastiani (2002:Section 2.1) gives a definition of text categorization, which can be summarized as the task of mapping a document onto a pre-determined set of classes. This is a very broad definition, and indeed one that is applicable to a wide variety of tasks, among which falls modern LangID. The archetypal text categorization task is perhaps classification of newswire articles according to the topics that they discuss, exemplified by the Reuters-21578 dataset (Debole and Sebastiani 2005) that is commonly used in text categorization research. However, LangID has particular characteristics that make it different from typical text categorization tasks:

1. Text categorization tends to use statistics about the frequency of words to model documents, but for LangID purposes there is no universal notion of a *word*: LangID must cater for languages where whitespace is not used to denote word boundaries.

2. In text categorization tasks, the set of labels usually only applies to a particular dataset. For example, it is not meaningful to ask which of the Reuters-21578 labels is applicable to the abstract of a biomedical journal article. However, in LangID there is a clear notion of language that is independent of domain; it is

possible to recognize that a text is in English regardless of whether it is from a microblog post or from a newspaper article.

3. In text categorization, the set of labels of interest is usually finite and predefined, but in LangID the set of languages is potentially open: the system may be required to identify that a text is from a language it does not have training data for.

4. In LangID, text in the same language can sometimes be written with different orthographies and stored in different encodings. Thus, despite belonging to a single logical class, texts from the same language may have very different representations under standard text categorization methods, and may require multiple non-overlapping models for each class.

5. Some text categorization methods can handle documents that are associated with multiple labels. In LangID, documents may be multilingual, in that they may contain text in more than one language. However, when this is the case the document can always be uniquely segmented into monolingual extents, which may be as small as individual words. This is in contrast to text categorization involving multi-labeled documents, where it is not necessarily possible to subdivide the document into specific extents associated with a single label.

These distinguishing characteristics present unique challenges and offer particular opportunities, so much so that research in LangID has generally proceeded independently of text categorization research. We will examine modern approaches to LangID in greater depth in Section 2.2, where we relate LangID to machine learning

in general, and examine the specific solutions that have been developed to tackle the difficulties presented by LangID. We will also provide a broader overview of open issues in LangID in Section 2.5, but in the next section we will focus specifically on the issues that we will tackle in the course of this thesis.

## 1.2 Main Contributions

In this section, we detail the specific issues that we have tackled as part of this thesis, focusing on the primary findings of the thesis. A more detailed synthesis of the open issues that have been identified by other researchers and the work done to address them is provided in Section 2.5. In contrast to this section, Section 1.3 provides an overview of the main narrative of the thesis without delving into details of the findings.

The first major issue we will tackle in this thesis is the variation present in a language across different sources of text, and its impact on the robustness of LangID systems when applied to text from various sources. Language varies substantially between different text sources, and any native speaker knows that, for example, the language used in a newspaper article looks very different from the language used in legal documents or on social media. This variation can be due to a wide variety of reasons, which we will examine detail in Section 3.1, and yet a native speaker is able to recognize his or her own language and understand the content of documents from a wide variety of sources. This indicates that there must be some shared properties across these sources that characterize a language *independently* of the source. In

Chapter 4, we investigate the cross-domain generalizability of LangID systems,[1] and quantify the loss in accuracy of LangID when data used to train a language identifier comes from a different source from the data the identifier is then applied to. In Chapter 5, we investigate the underlying causes of this decrease in accuracy, and develop a document representation for LangID that is robust to the variation in language between different sources of text.

The next issue that we will tackle is the assumption that a document contains text in a single language, sometimes referred to as the *monolinguality* assumption. As we discussed early in this chapter, work to date on LangID generally assumes that the entire document is written in a single language. *Multilingual* documents, i.e. documents that contain sections of text in more than one language, can exist for a variety of reasons, and detecting them can be useful in a variety of settings. We discuss the reasons and the settings in detail in Chapter 6, where we develop a method for LangID that is able to detect if a document is multilingual, identify the languages present and estimate the relative proportions of the document written in each language.

Finally, we address the issue of language identification in new and challenging domains. Whereas LangID research to date has tended to focus on LangID of longer, well-structured documents, many of the newer application areas for LangID have come about as a result of the growth in social media and the user-generated content it produces. Documents from such sources are typically short, and the language used is irregular, taking liberties with respect to conventional notions of spelling and

[1] We initially use the terms *domain* and *text source* interchangeably, providing a more specific definition of domain in Section 5.3.3.

grammar. Nonetheless, there is a clear need for effective language identification in such novel domains, for reasons such as user accessibility (e.g. identifying the language of a post in order to translate it into a language understood by the user), but also for data analytics reasons such as user profiling. In Chapter 7, we present a case study on language identification of short text messages from a microblogging service that was popular at the time of writing of this thesis.

## 1.3 Thesis Overview

We begin this thesis with a review of the relevant literature (Chapter 2), where we examine the common themes and ideas that underpin research in LangID. The literature review provides a brief history of LangID, tracing early work and identifying the first computational methods for detecting the natural language a document is written in. We then discuss modern approaches to LangID, highlighting the role of machine learning and comparing and contrasting relevant work on the different approaches taken towards issues of tokenization, feature selection, learning algorithms and empirical evaluation. Thereafter, we discuss some of the areas where LangID has been applied, followed by a broader discussion of open issues in LangID research and work to date on these areas, covering preprocessing, support for lower-density languages, "unseen" languages, multilingual documents, short texts, closely-related languages, issues of encoding, orthography and transliteration, non-linguistic metadata, and standardized corpora for LangID evaluation.

Chapter 3 discusses issues of data collection for LangID training and evaluation. It begins with a discussion of the role of variation in language between languages in

LangID and the importance of acquiring data to represent a wide scope of the possible variation in a language between different sources of text. We describe linguistic and non-linguistic ways that a language can vary between sources, and identify 9 data sources from which we acquire data for the experiments in this thesis. For each source, we give a brief description of the type and quantity of data available, as well as any characteristics or peculiarities of the source. We also give an overall summary of the datasets prepared for the experiments in Chapter 4 and Chapter 5.

Chapter 4 investigates the cross-domain generalizability of LangID systems. The main focus in this chapter is quantifying the effects of using training and test data from different sources on the accuracy of LangID. We identify three existing LangID systems that can be re-trained with new training data and describe each in detail, providing an analysis of how documents and languages are represented by each system, an explanation of the classification algorithm that each system implements, as well as details about any tunable parameters that each system has. We follow that with parameter tuning experiments to illustrate the effect that the tunable parameters have on the accuracy of each system. We then use the datasets described in Chapter 3 to assess accuracy of each system when training and test data are drawn from the same dataset, and plot learning curves to assess the impact of quantity of training data on each of the systems compared. We compare the *in-domain* results, where training and test data are drawn from the same text source, to *cross-domain* results, where the training and test data are drawn from different text sources. We provide comparisons in two distinct settings: (1) *single-source*, where training and test data are drawn from two different text sources, and (2) *all-source*, where the test data is

drawn from a single source, and the training data is the union of the data drawn from all the other sources combined. We compare the *in-domain* results to the single-source and all-source cross-domain results to quantify the impact on classification accuracy resulting from apply existing LangID systems trained on data from a different source to the target domain.

On the basis of the findings in Chapter 4, in Chapter 5 we take a closer look at the reasons for the difference in accuracy for a LangID system trained on data from the same source as the test data versus the same system trained on data from a different source or sources as the test data, and develop a strategy to mitigate the loss in performance due to training and test data coming from different sources. We begin with a more detailed comparison of the three systems we investigated in Chapter 4, showing how the systems share common concepts in the representation of documents and languages, and how each system implements supervised machine learning to determine the most likely language of a document. We discuss how the inductive learning hypothesis in machine learning relates to the concept of homogeneity from corpus linguistics. We use a standard method to evaluate the homogeneity with respect to language of documents from different datasets, and show how the pre-conditions of supervised machine learning are not met in the cross-domain classification settings examined in Chapter 4. We relate this to the study of transfer learning in the machine learning literature, and draw on a common theme in transfer learning to develop a feature selection methodology that takes into account both the language and the source of text, and show that this method yields a document representation that is more homogeneous with respect to language across different

datasets than the document representations used by the existing systems.

We then revisit the learning algorithms used by each system and identify the shared properties that make the algorithms suitable for LangID. We apply each algorithm to our novel document representation, which takes into account both the language and source of documents in its feature selection, and compare the accuracy of the resulting language identifiers to the existing systems re-trained on the same data. We conclude the chapter with an error analysis, examining the difference in accuracy between existing systems trained on in-domain data and our novel document representation using cross-domain data. We identify the major sources of error and discuss methods to minimize them.

In Chapter 6, we deal with LangID of *multilingual* documents, i.e. documents that contain text in more than one language. We describe a method that is able to detect documents that contain text in more than one language, while simultaneously determining the languages present as well as the relative proportions of each language in the document. This method builds on the work on document representation described in Chapter 5. It draws on work on generative mixture models that have become popular for text modeling tasks such as topic modeling. We show how our method relates to the naive Bayes learning algorithm described in detail in Chapter 5, as well as to the Latent Dirichlet Allocation model commonly used for topic modeling. We develop a synthetic dataset and use it to compare our method to other methods for LangID in multilingual documents that have been described in the literature. We also demonstrate the effectiveness of the method on a real-world dataset, on a task involving multilingual documents collected from a targeted web crawl for building text

corpora of lower-density languages. Finally, we integrate our work on multilingual documents with our work on cross-domain language identification from Chapter 5, and show that the document representation that we developed in Chapter 5 is able to better take advantage of training data from multiple different sources when combined with our method for LangID of multilingual documents.

Chapter 7 presents a case study in "off-the-shelf" LangID. In Chapter 2 we identify and describe a number of "off-the-shelf" LangID systems, i.e. complete software systems that are distributed with pre-trained models for a number of languages, such that end-users can run them as-is, without a need to provide their own training data. In Chapter 7, we benchmark these systems on user-generated messages on Twitter. Twitter is a popular microblogging service that has been explored by researchers in recent years due to the volume, variety and immediacy of the data available. One challenge in evaluating the accuracy of off-the-shelf LangID systems on Twitter messages is the lack of a broad-coverage dataset of language-labeled Twitter messages. We introduce a "mostly-automated" approach to constructing such a dataset, taking advantage of user identity to allow us to construct a corpus of language-labeled Twitter messages without using automated tools to directly determine the languages of the messages. We evaluate the accuracy of each off-the-shelf system using this new dataset, and analyze a number of simple techniques that have been proposed to improve the accuracy of off-the-shelf LangID on Twitter messages.

We conclude the thesis with Chapter 8, which summarizes the main findings and contributions of the thesis and provides a broad overview of directions that future work in LangID should take.

# Chapter 2

# Literature Review

As we discussed in Chapter 1, LangID is the task of determining the natural language(s) that a document (or part thereof) is written in. In Section 1.1, we discussed the relationship between LangID and more general approaches to text categorization, identifying the aspects that make the LangID task challenging and unique.

In this chapter, we will examine the common themes and ideas that underpin research in LangID. We begin with a brief history of research that has led to modern LangID (Section 2.1), and then proceed to review the literature, providing synthesis and analysis of existing research, focusing specifically on the representation of text (Section 2.2.1 and Section 2.2.2), the learning algorithms used (Section 2.2.3), and the methods for evaluating the quality of the systems (Section 2.2.4). We examine areas where LangID has been applied (Section 2.3), and then provide an overview of "off-the-shelf" LangID systems (Section 2.4). We conclude the chapter with a discussion of the open issues in LangID (Section 2.5), enumerating issues and existing efforts to address them.

## 2.1 A Brief History of LangID

Language identification as a task predates computational methods – the earliest interest in the area was motivated by the needs of translators, and simple manual methods were developed to quickly identify documents in specific languages, such as the use of particular diacritics (Newman 1987) or characteristic word tables (Ingle 1976). Language identification as a computational task has previously been attributed to Gold (1967) (Hughes *et al.* 2006; Trieschnigg *et al.* 2010; Chew *et al.* 2011), who sought to investigate language learnability from a language theory perspective. Gold (1967) gives a formalization of LangID as a closed-class classification problem, and investigates theoretical limitations on the ability to separate languages from different classes. However, the definition of language of Gold (1967) is only tangentially related to natural language. The results of Gold (1967) are much more general: the key result is that given sufficient labeled data in multiple languages (*language* as in *language theory* rather than *natural language*), it is possible to construct an algorithm that will correctly distinguish context-free languages. The proof given is a proof of existence; it does not actually suggest how to construct such an algorithm. In short, the results of Gold (1967) prove that (assuming documents are generated using a context-free grammar), there exists an algorithm to perform supervised text categorization that will converge to the correct answer in a finite number of steps. This may be a significant theoretical result, but it is a result that is generally taken for granted in any sort of supervised machine learning applied to text classification, and does not have any specific significance in the context of classifying documents by the natural language they are written in.

Much of the earliest work on automatic LangID was focused on identification of spoken language, or did not make a distinction between written and spoken language. The earliest reference to a computational method of distinguishing natural languages is perhaps the work of House and Neuburg (1977), which focuses on LangID of a spoken utterance, but their main contribution is simply to demonstrate the feasibility of LangID on the basis of a statistical model of broad phonetic information. They speculate that it is possible to use relative frequencies of particular phonetic patterns to automatically identify the language of a given utterance without the need for any manual intervention. However, their experiments do not use actual speech data, but rather "synthetic" data in the form of phonetic transcriptions derived from written text. Each text is mapped onto a 4-symbol alphabet, which consists of: (a) stop consonant, (b) fricative consonant, (c) non-vocalic sonorant, and (d) vowel. This phonetic transcription is then used to estimate the transition probabilities between symbols for each language. These models are then used to estimate the probability of an unseen text coming from each language, and the authors find that the "correct" model generally gives the highest probability. The authors used phonetic transcriptions of text as a close approximation of a representation that could be obtained by processing a speech waveform. However, in using written text as a proxy for speech, they inadvertently provide the first investigation of a computational method for LangID of text, using techniques that we will see again in Section 2.2.3.

The earliest work to describe a functional LangID program for text is perhaps Beesley (1988), which describes "Language Identifier: A Computer Program for Automatic Natural-Language Identification of On-line Text". The role of this program

was to route documents to machine translation systems, and the paper describes what has later come to be known as a byte $n$-gram model (Figure 2.1 on page 22). The key advance over manual methods of LangID used by translators, and a development that was foreshadowed by the work of House and Neuburg (1977) on statistical models based on phonetic classes, is the use of not just the binary presence/absence of particular letters or words, but rather the relative frequencies thereof. The fact that the distribution of letters is relatively consistent for a language was already well known, and is the basis of much of traditional cryptanalysis, but Beesley was the first to apply this directly to the task of identifying languages of written documents through a computer program.

A number of other earlier works on problems related to LangID are discussed by Muthusamy and Spitz (1997). Perhaps the most cited early work in automatic LangID of text is Cavnar and Trenkle (1994). We examine the method in much more detail in Chapter 4, but the gist of the method is to build per-document and per-language profiles, and classify a document by language according to which language profile it is most similar to, using a rank-order similarity metric defined by the authors. They evaluate their system on 3478 documents in 8 languages obtained from USENET newsgroups, reporting a best overall LangID accuracy of 99.8%. van Noord (1994) produced an implementation of the method of Cavnar and Trenkle (1994) named `TextCat`, which has become eponymous with the method itself. `TextCat` is packaged with pre-trained models for a number of languages, and so it is likely that the strong result reported by Cavnar and Trenkle (1994), combined with the ready availability of an "off-the-shelf" implementation has resulted in the exceptional popularity of

this particular method. Cavnar and Trenkle (1994) can be considered a milestone in automatic LangID, as it popularized the use of automatic methods on byte *n*-gram models for LangID, and to date the method is still considered a benchmark for automatic LangID (Kruengkrai *et al.* 2005; Tiedemann and Ljubešić 2012; Carter *et al.* 2013; Brown 2013).

## 2.2 Modern Approaches to Automatic LangID

In the previous section, we discussed early research that has shaped current approaches to automatic LangID. In this section, we focus on modern work, surveying contemporary literature on automatic LangID. Modern LangID systems can generally be decomposed into four key steps:

1. a representation of text is selected
2. a model for each language is derived from documents known to be written in each language
3. a function is defined that determines the similarity between a document and each language
4. the highest-scoring model determines the language of the document predicted by the system.

Very similar descriptions of the process can be found in House and Neuburg (1977), as well as Ueda and Nakagawa (1990:Section 5.1), and in a broad sense this describes a supervised machine learning approach (Section 5.3). In LangID, the choice of text

representation can usually be subdivided into two specific aspects: the tokenization of text (Section 2.2.1), and feature selection over the tokens (Section 2.2.2). The model of each language and the similarity function usually fall under the scope of the learning algorithm (Section 2.2.3), though the feature selection also has a part to play in defining the model of a language. Finally, determining suitable metrics for scoring models falls under the problem of empirical evaluation, discussed in Section 2.2.4.

Text representation deals with issues of transforming raw text into a form suitable for computationally processing a document, in our case to determine the natural language that the document is written in. The representations used in work to date come under a variety of descriptions, including phonetic classes (House and Neuburg 1977), letter sequences (Beesley 1988), mixture of byte $n$-grams (Cavnar and Trenkle 1994), character shape codes (Sibun and Spitz 1994), bigraph/trigraph frequencies (Souter *et al.* 1994), trigram models (Grefenstette 1995), grammatical words (Giguet 1995), Markov processes (Dunning 1994), variable length $n$-grams (Cowie *et al.* 1999), grammatical-class models (Dueire Lins and Gonçalves 2004), symbol features (Xafopoulos *et al.* 2004), and compressive models (Teahan 2000; Yamaguchi and Tanaka-Ishii 2012).

Due to the lack of standardized datasets and evaluation metrics in LangID research (which we discuss in more detail in Section 2.2.4), it is very difficult to contrast the relative effectiveness of the different approaches to text representation. Results across different datasets are generally not comparable, as a method's efficacy can vary substantially with parameters such as the number of languages considered, the relative amounts of training data, and the length of the test documents (Baldwin

and Lui 2010a). As we will see in the following sections, the different document representations are very similar in terms of the underlying ideas, and can be seen as different parametrizations of the same basic process. Where possible, we will compare and contrast results from work to date on different aspects of this process.

Table 2.1 provides an overview of experimental configurations that have been explored in the literature. Each publication is summarized in terms of the algorithm(s) explored, the granularity of text used, the source of the text and the document representation. Where possible, the authors' own descriptions have been used to populate the "Algorithm" and "Representation" columns. This makes the algorithms and representations appear more diverse than they actually are, which we discuss in more detail in the respective sections on algorithms (Section 2.2.3) and representation (Section 2.2.1). An entry of "multiple" indicates that the authors experiment with different algorithms and/or representations in the paper, usually to compare and contrast between them. In the "Algorithm" column, many publications are listed as using "fingerprinting". This is a catch-all term we introduce to describe methods that construct representations of each language that are specific to that publication, without giving an explicit name to the method. In the granularity column, "snippet" refers to publications that sample sections of a document. These sections are usually of fixed length, either in terms of words or characters, and the same publication often uses multiple lengths to compare accuracy. In the "Source" column, "various" indicates that text from more than one source has been used.

| **Reference** | **Algorithm** | **Granularity** | **Source** | **Representation** |
|---|---|---|---|---|
| Cavnar and Trenkle (1994) | rank-order statistics | document | newsgroup | byte $n$-gram |
| Dunning (1994) | Markov process | snippet | parallel text | byte seq |
| Sibun and Spitz (1994) | discriminant analysis | scanned text | not specified | word shape |
| Souter *et al.* (1994) | fingerprinting | document | linguistic resource | bigram/trigram |
| Combrinck and Botha (1995) | fingerprinting | document | not specified | letter trigram |
| Grefenstette (1995) | fingerprinting | sentence | linguistic resource | trigram/short word |
| Kikui (1996) | fingerprinting | webpage | web crawl | byte $n$-gram |
| Sibun and Reynar (1996) | relative entropy | document | linguistic resource | character shape |
| Adams and Resnik (1997) | conditional probability | document | parallel text | character $n$-gram |
| Elworthy (1998) | naive Bayes | snippet | linguistic resource | word shape |
| Cowie *et al.* (1999) | fingerprinting | snippet | not specified | variable length $n$-gram |
| Prager (1999a) | vector space | document | web document | byte $n$-gram/short words |
| Hakkinen and Tian (2001) | decision trees | place names | name database | letter seq |
| Poutsma (2002) | fingerprinting | document | linguistic resource | sampled word/$n$-grams |
| Tian and Suontausta (2003) | neural network | document | not specified | word counts |
| Dueire Lins and Gonçalves (2004) | fingerprinting | document | web document | word class |
| Padró and Padró (2004) | multiple | document | newspaper | character seq |
| Takçi and Soğukpınar (2004) | centroid-based | document | web document | letter counts |
| Xafopoulos *et al.* (2004) | hidden Markov models | document | web document | symbol features |
| Kruengkrai *et al.* (2005) | fingerprinting/SVM | document | news articles | string kernel |
| Martins and Silva (2005) | fingerprinting | document | web document | character $n$-gram |
| McNamee (2005) | fingerprinting | sentence | various | common word counts |
| Tran and Sharma (2005) | Markov model | snippet | web document | letter seq |
| Windisch and Csink (2005) | fingerprinting | snippet | translations | descriptive statistics |
| Artemenko *et al.* (2006) | multiple | snippet | newspaper | character $n$-gram/words |
| Da Silva and Lopes (2006) | covariance similarity | snippet | legal text | discriminant seq |
| Mandl *et al.* (2006) | multiple | snippet | newspaper | multiple |
| Murthy and Kumar (2006) | multiple linear regression | snippet | not specified | akshara seq |
| Singh (2006) | mutual cross entropy | snippet | various | byte $n$-gram |
| Vojtek and Bieliková (2007) | Markov process | document | newswire | character seq |
| Grothe *et al.* (2008) | rank-order statistics | document | various | multiple |
| Choong *et al.* (2009) | boolean $n$-gram matching | document | newspaper | byte $n$-gram |
| Rehurek and Kolkus (2009) | word relevance | sentence | Wikipedia | character $n$-gram |
| Baldwin and Lui (2010a) | multiple | document | various | multiple |
| Pienaar and Snyman (2010) | spell checking | document | government web | individual words |
| Trieschnigg *et al.* (2010) | multiple | document | linguistic resource | multiple |
| Vatanen *et al.* (2010) | multiple | short text | UDHR | character $n$-gram |
| Yang and Liang (2010) | search engine | document | Wikipedia | character $n$-gram |
| Chew *et al.* (2011) | fingerprinting | document | various | byte seq |
| Lui and Baldwin (2011) | naive Bayes | document | various | byte $n$-gram |
| Ng and Selamat (2011) | optimum profile | document | web document | character $n$-gram |
| Stupar *et al.* (2011) | hybrid | doc/para | web document | function word + letter seq |
| Tromp and Pechenizkiy (2011) | graph-based | message | Twitter | character $n$-gram |
| Winkelmolen and Mascardi (2011) | naive Bayes | short text | subtitles | character $n$-gram |
| Bergsma *et al.* (2012) | multiple | message | Twitter | char $n$-gram + metadata |
| Botha and Barnard (2012) | multiple | snippet | various | character $n$-gram |
| Brown (2012) | vector space | sentence | various | byte $n$-gram |
| Lui and Baldwin (2012) | naive Bayes | document | various | byte $n$-gram |
| Majliš (2012) | multiple | snippet | Wikipedia | byte $n$-gram |
| Milne *et al.* (2012) | multiple | document | various | word + character $n$-gram |
| Takçi and Ekinci (2012) | multiple | snippet | linguistic resource | weighted letters |
| Takçı and Güngör (2012) | centroid-based | snippet | linguistic resource | individual characters |
| Vogel and Tresner-Kirsch (2012) | graph-based | message | Twitter | character $n$-gram |
| Brown (2013) | k-nearest neighbor | sentence | various | byte $n$-gram |
| Carter *et al.* (2013) | fingerprinting | message | Twitter | char $n$-gram + metadata |
| Goldszmidt *et al.* (2013) | multiple | message | various | character/word $n$-gram |
| Brown (2014) | multiple | sentence | various | byte $n$-gram |
| Lui and Baldwin (2014) | multiple | message | Twitter | multiple |
| Simões *et al.* (2014) | neural network | document | various | character class + trigrams |

Table 2.1: Summary of experiment configurations.

| | |
|---|---|
| **1-gram** | l, a, n, g, u … |
| **2-gram** | la, an, gu, ua, ag … |
| **3-gram** | lan, ang, gua, uag, age … |
| **4-gram** | lang, angu, guag, uage, age_ … |
| **5-gram** | langu, angua, guage, uage_, age_i … |

Figure 2.1: Character $n$-gram representations of the string "language identification".

### 2.2.1 Tokenization

The first step in representing a document for purposes of LangID deals with the issue of *tokenization*, that is how the continuous stream of characters that comprises text should be divided into units that are meaningful for distinguishing between languages.

In computational processing of written text, a distinction is generally drawn between character-oriented and word-oriented models (Kay 1997). Word-oriented models have a rich history in text processing, and underpin most work in information retrieval (van Rijsbergen 1979; Witten *et al.* 1999) and text categorization (Sebastiani 2002). A model that represents a document as a distribution over the frequency of the words contained is colloquially referred to as a bag-of-words model.[1]

[1]The origins of the term "bag-of-words" is unclear. An early use of the phrase found in books digitized by Google uses the term as an insult (Jortin and Clerc 1808:p.p.398), wherein Hieronymus Emserus is described as "an impertinent prater, *saccum verborum*, a mere bag of words". The use of "bag-of-words" to describe language starts to appear in the 1970s. Moulton (1975:p.p.18) discusses how some view language as "little more than a bag of words", and Goodman (1973:p.p.11) describes how, in the methodology being developed, "language is seen [$\cdots$] as much more than the bag of words we used to think it was." Similarly, Harris (1970:p.p.785) argues that "language is not merely a bag of words but a tool with particular properties which have been fashioned over the course of its use." Perhaps somewhat ironically given the assertions of Harris and Goodman, the modern use

Character-oriented models for LangID generally involve frequency counts over specific character sequences, often referred to as character $n$-grams or simply $n$-grams; an illustrative sample of character $n$-grams is given in Figure 2.1. The use of the term $n$-gram to refer to character sequences sometimes causes confusion, as in empirical approaches to computational linguistics, an $n$-gram usually implicitly refers to a *word* $n$-gram. Suzuki *et al.* (2002) proposed the use of the term *shift-codon* as an alternative name for such character sequences, but this term has not seen wider uptake.

Character-oriented models tend to be more commonly used in LangID. Despite the varied terminology used to describe them, most character-oriented representations share some fundamental properties. Documents are viewed as a stream or sequence of characters (Cavnar and Trenkle 1994; Kikui 1996), and this stream is processed in a number of ways to arrive at a concrete document representation, which is generally a function of the relative frequencies of particular character sequences. These sequences are selected such that their relative distributions are expected to differ greatly between languages. The most significant difference between word-level and character-level models is that in character-level models, sequences of characters that are adjacent and overlap are counted separately. For example, the word *language* would produce a single count in a word-level model, but would produce counts for *lang*, *angu*, *ngua*,

of the "bag-of-words" models language as being just that – a bag full of words, as if a document had been cut up into individual words that have been all dumped into a bag. From a probabilistic perspective, the bag-of-words model of a document is essentially the probability of drawing any given word out of this bag at random. In this context, the use of the term "bag" is related to the mathematical entity known as the *multiset*, and indeed Knuth (1998:p.p.636) notes that "bag" is an alternative name proposed for the concept.

*guag* and *uage* in a character 4-gram model. This overlap can lead to redundant information, and may not correspond to independence assumptions between features made by certain learning algorithms. On the other hand, the breaking of words into smaller segments provides an automatic means of accessing linguistically-motivated features such as distinctive prefixes or suffixes, which may be much more common than any individual word using them. One question in using character-sequence models is whether to allow sequences to span across whitespace. Some authors choose to do so (Grefenstette 1995; Brown 2013), whereas others enforce that the sequences may only start or end with whitespace but not contain it (Cavnar and Trenkle 1994).

Where word oriented-models are used for LangID, word segmentation is usually done by simply tokenizing on whitespace (e.g. McNamee (2005)), which limits the applicability of such methods to languages where words are whitespace-delimited. Some authors have tested word-oriented models alongside character-oriented ones (Grefenstette 1995; Poutsma 2002), with mixed conclusions. Word-oriented models have had particular success in discriminating closely-related languages. For example, the model used by Tiedemann and Ljubešić (2012) to discriminate between Bosnian, Serbian and Croatian makes use of word frequencies, focusing in particular on identifying words that are *not* valid in a particular language. Zampieri (2013) also uses a bag-of-words model to distinguish between continental and colonial varieties of French, Spanish and Portuguese. Some authors have also used document representations that mix word and character $n$-gram representations (Prager 1999a). Interesting variations on word-oriented representations include word shape representations (Sibun and Reynar 1996; Elworthy 1998), which are intended for LangID of images of text as part of an op-

tical character recognition (OCR) process, and word-class models (Dueire Lins and Gonçalves 2004), which map words onto grammatical classes. An interesting midpoint between word-oriented and character-oriented models is syllable-oriented models for Indian languages. Murthy and Kumar (2006) argue that fundamental unit of writing in Indian scripts is a group of characters known as an *akshara*, and that akshara sequences are appropriate for modeling Indian languages.

Work to date that has compared character-oriented and word-oriented models for LangID has generally found that character-oriented models are more accurate (Souter *et al.* 1994; Prager 1999a). Furthermore, Prager (1999a) found that a combined representation using character and word features had higher accuracy than either feature set alone. One area where word-centric models have been found to outperform character-oriented models is in the discrimination of closely-related languages (see Section 2.5.6), because the subtle differences are lost in the character $n$-gram "wash". Another reason why character-oriented models tend to be more popular is that word segmentation is not trivial in languages that are morphologically complex, or that do not use word delimiters. Character-oriented models are preferred when such languages are included due to their universal applicability to text in any language.

It is not clear from research to date if there is some inherent universally-optimal value for $n$, the length of character sequences to use. Some authors have considered only a single value for $n$, such as Takçi and Ekinci (2012) and Takçı and Güngör (2012), which use the frequencies of single letters (i.e. $n = 1$), and Grefenstette (1995) and Suzuki *et al.* (2002), which only tests $n = 3$. Others test multiple discrete values of $n$, such as Prager (1999a), which tests $n = 2, 3, 4, 5$ and finds that $n = 4$

is optimal. Majliš (2012) tests $n = 1, 2, 3, 4$, and finds that $n = 2$ is optimal for standard classification algorithms, whereas $n = 4$ is optimal for algorithms developed specifically for LangID. Another possibility is to use a range of values for $n$, such as in Cavnar and Trenkle (1994), where features are a mixture of $n$-grams of length 1–4. Studies that have considered values of $n \geq 5$ have generally reported that the optimal value for $n$ is 3 or 4 (Adams and Resnik 1997; Prager 1999a; Choong *et al.* 2009; Vojtek and Bieliková 2007; Baldwin and Lui 2010a). Indeed, in contrast to Takçi and Ekinci (2012); Takçı and Güngör (2012), Brown (2012) explicitly dismisses $n = 1, 2$ as insufficiently informative for LangID. Related research in text retrieval has found that $n = 4$ is a good choice for European languages (McNamee and Mayfield 2004). One possible explanation for this optimal value is that $n$ =3–4 very roughly corresponds to the average size of a morpheme in many languages, and so captures characteristics such as distinctive prefixes or suffixes. An exception to this trend is Brown (2012) which reports the best results for $n = 6$, with reduced accuracy for higher values of $n$. This may be due to Brown (2012) considering more languages simultaneously, which may require longer sequences to discriminate between similar languages, consistent with the findings of Tiedemann and Ljubešić (2012) which find that word-oriented models are better for discriminating between Bosnian, Serbian and Croatian than character-oriented models.

A subtlety in character-oriented models is what exactly is considered a character, where a distinction needs to be made between individual symbols used by a language, and the underlying digital representation in terms of a sequence of bytes. Each abstract symbol (e.g. a letter in an alphabet, or a Chinese ideogram) is represented

by a codepoint, which is a number that indexes the specific symbol in a *character set*. The mapping between the codepoint and the actual sequence of bytes that is stored and transmitted by computers is known as an *encoding*. Well-known and commonly-used encodings include ASCII, Latin-1 and UTF-8. For some encodings there is a one-to-one mapping between bytes and codepoints (e.g. in ASCII, characters are always represented by exactly one byte). Where this is the case, there is no practical difference between a byte-oriented model and a character-oriented model (a point noted by Singh (2006)), and so much early work on European languages does not make any distinction between the two. A further point of distinction needs to be drawn between character-oriented and letter-oriented models, where the latter is sometimes used to mean models that only consider a subset of the possible characters, such as in the case of Hakkinen and Tian (2001), which only consider the 26 letters of the English alphabet and discard all other characters. Symbol features (Xafopoulos *et al.* 2004) have also been proposed, which cover the letters of several alphabets as well as commonly used symbols.

One issue that can arise in character-oriented models for broad sets of languages is data sparsity (Simões *et al.* 2014), resulting from certain languages using a large variety of symbols, of which only a small proportion will be present in any given document. One proposed solution is "codeplane reduction" (Nakatani 2010b), where codepoints from specific Unicode blocks such as punctuation, as well as more language-specific blocks such as Japanese-language Hiragana/Katakana script, are mapped to a single codepoint used to represent the entire block. Simões *et al.* (2014) use a similar process, where characters are mapped to character classes roughly according to their script.

Baldwin and Lui (2010a) compared representations based on byte and codepoint $n$-grams, and found that byte-oriented models generally attained better accuracy than codepoint-oriented models. Approximating a stream of characters (or codepoints) with an underlying stream of bytes presents two main challenges: (1) some encodings represent certain characters using multi-byte sequences, and these sequences can be of variable length (e.g. UTF-8); and (2) some languages have several common encodings in use (e.g. popular encodings for Chinese include GuoBiao, Big5 and UTF-8). However, the interaction between encoding and LangID is a relatively under-researched area, which we discuss is more detail in Section 2.5.7.

### 2.2.2 Feature Selection

From a theoretical perspective, the document representation consists of a distribution over the entire space of possible character sequences, whether this space is the space of words or the space of fixed-length sub-sequences. In practice, such a space is either exponentially large (in the case of character $n$-grams), or infinite (in the case of words), which presents computational challenges. The practical solution to this is to select a subset of sequences which we will consider as being "relevant" to discriminating between languages, a process known as feature selection. Feature selection is a well-studied problem, and it provides benefits beyond complexity reduction. We discuss theoretical aspects further in Section 5.4. In this section, we limit ourselves to the discussion of how work to date has implemented feature selection – the automatic identification of relevant character sequences – for purposes of LangID. In some methods the relevant character sequences

are externally specified. Examples include the use of specific words (Giguet 1995; Dueire Lins and Gonçalves 2004), or word fragments thought to be characteristic of particular languages (Dunning 1994). As an alternative to externally-specified sequences, the relevant sequences can be learned from labeled training data (noting that this is a distinct problem from learning a model of a language, the subject of Section 2.2.3). In such approaches, the character stream is converted to a frequency count over character sequences. The individual sequences derived are referred to as *character* $n$-grams; an illustrative example is given in Figure 2.1 on page 22. The character $n$-gram representation allows the characteristic sequences of each language to be learned directly from the data. Examples of this in the literature include the use of feature selection methods such as raw frequency counts (Cavnar and Trenkle 1994; Grefenstette 1995), measures of discriminant ability (Da Silva and Lopes 2006), as well as methods based on information gain (Lui and Baldwin 2011). Brown (2013) introduces a heuristic method to reduce the overlap in the set of sequences. Overlapping sequences are compared for frequency, and if the shorter sequence occurs with roughly the same frequency as the longer sequence, the shorter one is eliminated. In these approaches, the total set of possible $n$-grams is reduced to a representative set, which has the advantage of making the model smaller, in turn requiring reduced computational resources. Another advantage is that feature selection can help to reduce "overfit" (discussed in more detail in Section 5.4).

In contrast, methods such as Markov processes (Dunning 1994; Vojtek and Bieliková 2007) and compressive models (Teahan 2000; Yamaguchi and Tanaka-Ishii 2012) do not have a pre-defined set of "relevant" sequences, but instead they model the prob-

ability of arbitrary sequences of characters. In practice, this turns out to be very similar to estimating the relative distribution of pre-determined sequences. Consider a first-order Markov process for generating sequences of characters, where the next character generated is conditioned only on the previous character. The maximum likelihood estimate for the transition probabilities of the process corresponds exactly to a renormalized frequency count over character 2-grams, and indeed this intuition can be generalized to $n$-gram sequences of any order. Essentially, models that have an explicit set of character sequences $S$ can be interpreted as a probabilistic language model defined as follows:

$$P(s) = \begin{cases} \frac{N_s}{\alpha + \sum_{i \in S} N_i}, & \text{if } s \in S. \\ \alpha_s \approx 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

In Equation 2.1, $N_i$ is the frequency with which sequence $i$ occurs, $\alpha$ is a small amount of probability mass set aside to be distributed over sequences that do not occur in the training data, and $\alpha_s$ is a small value may be determined by some method of smoothing or interpolation. Adams and Resnik (1997) experiment with add-$k$ smoothing and Good-Turing smoothing. Brown (2013) uses a different form of smoothing, where the language prediction for previous sentences is used as a prior for subsequent sentences. Brown (2014) proposes to apply "non-linear mappings" to each $P(s)$ for a given language, in the form of: (1) raising each $n$-gram probability for a given language to a power $\gamma < 1$; and (2) a normalized variant of the logarithm function, again with an exponential parameter $\tau$. He finds that the mappings improve the accuracy of a range of existing language identifiers, over two corpora. Giwa and Davel (2013) test various methods for smoothing, and report that any form of smoothing

substantially improves accuracy of LangID for individual words as compared to a baseline naive Bayes model.

### 2.2.3 Learning Algorithms

In the previous sections, we compared how text is represented in work to date, examining issues of tokenization and of feature selection. In this section, we contrast the different approaches to building models of languages that can be used to determine what language a document is written in. Akin to the apparent diversity in document representation, there is a corresponding diversity in the descriptions of the learning algorithms applied to induce language classifiers. Many learning algorithms from the machine learning literature have been applied in some form to the task of LangID, including support vector machines (Kruengkrai *et al.* 2005; Majliš 2012; Takçi and Ekinci 2012), neural networks (Tian and Suontausta 2003; Sagiroglu *et al.* 2007; Takçi and Ekinci 2012; Simões *et al.* 2014), decision trees (Hakkinen and Tian 2001), vector-space models (Prager 1999a; Takçi and Soğukpınar 2004; Brown 2013), and naive Bayes models (Grefenstette 1995; Hakkinen and Tian 2001; Lui and Baldwin 2011; Winkelmolen and Mascardi 2011).

Many of the learning algorithms applied to LangID can be understood in the framework of Bayesian classification, in which we compute $P(L_i|D)$, the probability of a given language $L_i$ from a closed set of candidate languages $L$ given a particular document $D$. The identified language $l$ of document $D$ is thus determined as the most likely language conditioned on the document $D$ (Equation 2.2).

$$l = \text{argmax}_{L_i \in L} P(L_i|D) \tag{2.2}$$

Bayes' theorem allows us to re-express the likelihood of the language given the document ($P(L_i|D)$) in terms of the product of the likelihood of the document given the language ($P(D|L_i)$) and the prior probability of $L_i$ ($P(L_i)$), normalized by the document probability $P(D)$ (Equation 2.3).

$$l = \text{argmax}_{L_i \in L} \frac{P(D|L_i)P(L_i)}{P(D)} \tag{2.3}$$

Since $P(D)$ is independent of $L_i$, it does not affect the relative ordering of languages and thus can be dropped for purposes of determining the most likely language (Equation 2.4).

$$l = \text{argmax}_{L_i \in L} P(D|L_i)P(L_i) \tag{2.4}$$

Implementing a Bayesian classifier thus requires methods for estimating the likelihood of a document given a particular model of a language ($P(D|L_i)$), as well as the prior probability over the set of languages ($P(L_i)$). Methods differ in how they estimate these two quantities. Approaches to computing $P(D|L_i)$ include Markov processes (Dunning 1994; Vojtek and Bieliková 2007), naive Bayes methods (Grefenstette 1995; Elworthy 1998; Hakkinen and Tian 2001; Lui and Baldwin 2011; Winkelmolen and Mascardi 2011), and compressive models (Teahan 2000). Language identifiers based on neural networks can also be understood in this context, as each node in the output layer effectively computes the likelihood of the input under the class modeled by that particular node (Tian and Suontausta 2003).

Where $P(L_i)$ is estimated, it is normally by maximum likelihood methods (Lui and Baldwin 2011). However, it is also common to assume a uniform prior (Dunning 1994; Grefenstette 1995; Kikui 1996; Elworthy 1998; Teahan 2000). A uniform prior encodes the notion that no assumptions are made about what languages a document

is most likely to be written in – without seeing a document, it is considered to be equally likely that the document is written in any of the languages the classifier knows about. Depending on the application, this may or may not be a desirable characteristic of the classifier. Another characteristic of Bayesian methods is that, under the assumption that the input document is written in a single language, it is possible to determine when sufficient evidence to make a decision has been collected and thus avoid processing the rest of the document (Elworthy 1998; Nakatani 2010b).

Another group of methods can be summarized as language fingerprinting approaches, which are also known as language profiling, nearest prototype (in contrast to nearest-neighbor), or Rocchio-style (Rocchio 1971) methods. Fingerprinting methods construct a single "fingerprint" for each language, using information such as the relative frequency of particular sequences of characters, and classify documents by finding the most similar fingerprint. More formally, fingerprinting methods have a fingerprinting function $f$ that maps the set of documents from a given language $L_i \in L$ onto a single per-language pseudo-document $S_{L_i}$, and a distance metric $m$ used to compare a document to be classified to each pseudo-document. The identified language $l$ of a document $D$ is thus selected as the language of the pseudo-document $S_{L_i}$ that $D$ is most similar to, as measured by $m(D, S_{L_i})$. This is summarized in Equation 2.5.

$$l = \text{argmin}_{L_i \in L}\, m(D, S_{L_i}) \tag{2.5}$$

The fingerprinting function is usually simple, such as the sum (Cavnar and Trenkle 1994) or the average (Prager 1999a; Takçi and Soğukpınar 2004) across document vectors, which consists of counts across a specific set of features such as short words

(Grefenstette 1995; Prager 1999a) or letter sequences (Cavnar and Trenkle 1994; Souter *et al.* 1994). This produces per-language pseudo-documents that can be thought of as a "typical" document in the given language. The per-language pseudo-documents can be interpreted in several ways, depending on the machine learning paradigm we view them from. From the perspective of vector-space models, the pseudo-document is a centroid of the cluster of points associated with a particular language. From a probabilistic perspective, if we interpret each document as an independent and identically distributed sampling from a multinomial distribution over byte $n$-grams, the pseudo-documents produced by averaging the document vectors corresponds to the maximum likelihood estimate of the parameters of the underlying multinomial distribution.

The varying interpretations of the pseudo-document lead us to a variety of ways to measure the similarity between a document and the per-class pseudo-documents. These include metrics based on rank order statistics (Cavnar and Trenkle 1994; Goldszmidt *et al.* 2013), Markov processes (Dunning 1994), information theory (Sibun and Reynar 1996; Baldwin and Lui 2010a), string kernels (Kruengkrai *et al.* 2005) and vector space models (Prager 1999a; Takçi and Soğukpınar 2004; McNamee 2005; Brown 2013).

It is difficult to determine which learning algorithm is best for LangID independently of the document representation used. Studies that have attempted to do so have arrived at contrasting conclusions. Vojtek and Bieliková (2007) empirically compared the methods proposed by Dunning (1994) and Teahan (2000) using data from 8 European languages, and found that their accuracy was very close. Baldwin and Lui

(2010a) compared naive Bayes, k-Nearest Neighbor (k-NN) and support vector machines over a standardized document representation based on byte $n$-grams of fixed order, and found that a cosine-based 1-NN model performed best. However, when a mixture of different orders of $n$-grams (Cavnar and Trenkle 1994) was used, a 1-NN model based on skew divergence (Lee 1999) performed best. The out-of-place distance metric proposed by Cavnar and Trenkle (1994) was also tested with 1-NN but performed worse than skew divergence. Majliš (2012) considered 5 algorithms, and found that support vector machines (SVM) achieved the best accuracy overall. Mandl *et al.* (2006) compared vector space models, the out-of-place metric, naive Bayes, and word-based models and found that naive Bayes methods had the lowest error rate. Finally, Goldszmidt *et al.* (2013) also considered multiple algorithms under a standardized document representation. They found that the best performance was obtained using "Spearman's Footrule", which is identical to the out-of-place distance metric described in Cavnar and Trenkle (1994). The only conclusion we can draw from this is that the target domain is clearly a factor in determining which learning algorithm is best for LangID. We examine this issue and other issues in systematic evaluation of LangID more closely in the following section.

### 2.2.4 Empirical Evaluation

In the previous two sections, we have alluded to issues of evaluation in LangID research to date. In this section, we examine the literature more closely, providing a broad overview of the metrics that have been used, as well as the experimental settings in which LangID research has been evaluated.

| **Reference** | # Lang | Overall | | Per-Language | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Acc | Err | P | R | F | CM | LC | TS | S | NF |
| Cavnar and Trenkle (1994) | 8 | ✓ | | | ✓ | | | | ✓ | | ✓ |
| Dunning (1994) | 2 | | ✓ | | | | | | | | |
| Sibun and Spitz (1994) | 23 | | | | | | ✓ | | | | |
| Souter *et al.* (1994) | 9 | ✓ | | | | | | ✓ | | | |
| Combrinck and Botha (1995) | 12 | | | | | | | ✓ | ✓ | | |
| Grefenstette (1995) | 9 | | | | ✓ | | ✓ | | ✓ | | |
| Kikui (1996) | 9 | | ✓ | | | | ✓ | | | | |
| Sibun and Reynar (1996) | 27 | ✓ | | | | | ✓ | | | | |
| Adams and Resnik (1997) | 2 | ✓ | | | ✓ | | | | | | |
| Elworthy (1998) | 18 | ✓ | | | | | ✓ | | ✓ | | |
| Cowie *et al.* (1999) | 34 | | ✓ | | | | | | ✓ | | |
| Prager (1999a) | 13 | | | | ✓ | | | | ✓ | | |
| Hakkinen and Tian (2001) | 4 | ✓ | | | ✓ | | | | | | |
| Poutsma (2002) | 10 | ✓ | | | | | | | ✓ | ✓ | |
| Tian and Suontausta (2003) | 25 | ✓ | | | | | | | | | |
| Dueire Lins and Gonçalves (2004) | 4 | | | | ✓ | | | | | | |
| Padró and Padró (2004) | 6 | ✓ | | | | | | ✓ | ✓ | | |
| Takçi and Soğukpınar (2004) | 4 | | | | ✓ | | | | | | |
| Xafopoulos *et al.* (2004) | 5 | ✓ | | | | | ✓ | | | | |
| Kruengkrai *et al.* (2005) | 17 | ✓ | | | | | | | | | |
| Martins and Silva (2005) | 12 | | | | ✓ | | ✓ | | | | |
| McNamee (2005) | 10 | | | ✓ | | | ✓ | | | | |
| Tran and Sharma (2005) | 7 | | ✓ | | | | ✓ | | ✓ | | |
| Windisch and Csink (2005) | 5 | | | | ✓ | | ✓ | | ✓ | | |
| Artemenko *et al.* (2006) | 6 | | ✓ | | | | | | ✓ | | |
| Da Silva and Lopes (2006) | 19 | | | | | | ✓ | | | | |
| Mandl *et al.* (2006) | 8 | | ✓ | | | | | | ✓ | | |
| Murthy and Kumar (2006) | 9 | | | ✓ | ✓ | ✓ | | | | | |
| Singh (2006) | 39 | ✓ | | | | | | | ✓ | | |
| Vojtek and Bieliková (2007) | 8 | | | | ✓ | | | ✓ | | | |
| Grothe *et al.* (2008) | 9 | ✓ | | | | | | | | | |
| Choong *et al.* (2009) | 68 | ✓ | | | | | | | | | |
| Rehurek and Kolkus (2009) | 9 | | | ✓ | ✓ | | | | | | |
| Baldwin and Lui (2010a) | 67 | ✓ | | ✓ | ✓ | ✓ | | ✓ | ✓ | | |
| Pienaar and Snyman (2010) | 11 | ✓ | | | | | ✓ | | | | |
| Trieschnigg *et al.* (2010) | 16 | | | ✓ | ✓ | ✓ | | | | | |
| Vatanen *et al.* (2010) | 281 | | | ✓ | ✓ | | | | | | ✓ |
| Yang and Liang (2010) | 12 | ✓ | | | | | | | | | |
| Chew *et al.* (2011) | 182 | ✓ | | | | | | | | | |
| Lui and Baldwin (2011) | 89 | ✓ | | | | | | | | ✓ | |
| Ng and Selamat (2011) | 23 | ✓ | | | | | ✓ | | | | |
| Stupar *et al.* (2011) | 12 | ✓ | | | | | | | | | |
| Tromp and Pechenizkiy (2011) | 6 | ✓ | | | | | | | | | |
| Winkelmolen and Mascardi (2011) | 22 | | ✓ | | | | | | ✓ | | |
| Bergsma *et al.* (2012) | 12 | ✓ | | | | | | | | | |
| Botha and Barnard (2012) | 11 | ✓ | | | | | ✓ | ✓ | | | |
| Brown (2012) | 923 | | ✓ | | | | | ✓ | | | ✓ |
| Lui and Baldwin (2012) | 67 | ✓ | | | | | | | | ✓ | |
| Majliš (2012) | 90 | ✓ | | | | | | ✓ | ✓ | ✓ | ✓ |
| Milne *et al.* (2012) | 6 | | | | ✓ | | ✓ | | ✓ | | |
| Takçi and Ekinci (2012) | 9 | | | | ✓ | | | | | ✓ | |
| Takçı and Güngör (2012) | 9 | ✓ | | ✓ | ✓ | ✓ | | | | ✓ | |
| Vogel and Tresner-Kirsch (2012) | 6 | ✓ | | ✓ | ✓ | ✓ | | | | | |
| Brown (2013) | 1100 | | ✓ | | | | | ✓ | | ✓ | |
| Carter *et al.* (2013) | 5 | ✓ | | | ✓ | | | | | | |
| Goldszmidt *et al.* (2013) | 52 | ✓ | | | | | | | ✓ | | ✓ |
| Brown (2014) | 1311 | | ✓ | | | | | | | | ✓ |
| Lui and Baldwin (2014) | 65 | ✓ | | ✓ | ✓ | ✓ | | | | | |
| Simões *et al.* (2014) | 25 | ✓ | | | | | | | | | |

Table 2.2: Summary of empirical evaluations. Acc=Accuracy, Err=Error Rate, P=Precision, R=Recall, F=F-score, CM=Confusion Matrix, LC=Learning Curve, TS=Test Size, S=Speed, NF=Number of Features.

Table 2.2 summarizes some of the parameters of empirical evaluations in work to date. The most common approach is to treat the task as a document-level classification problem. Given a set of evaluation documents, each having a known correct label from a closed set of labels (often referred to as the "gold-standard"), and a predicted label for each document from the same set, the document-level accuracy is the proportion of documents that are correctly labeled over the entire evaluation collection. This is the most often-reported metric, and conveys the same information as the error rate, which is simply the proportion of documents that are incorrectly labeled (i.e. $1 - \text{accuracy}$).

Authors sometimes provide a per-language breakdown of results. There are two distinct ways in which results are generally summarized per-language: (1) precision, in which documents are grouped according to their predicted language; and (2) recall, in which documents are grouped according to what language they are actually written in. More formally, consider a set of documents $D = \{D_1 \cdots D_m\}$ and a set of languages $L = \{L_1 \cdots L_n\}$. For each document $D_x$ we denote that the document is written in language $L_y$ by $D_x \rightarrow L_y$, and that the system predicts the document is written in $L_z$ by $D_x \triangleright L_z$. We use an overline to denote negation, for example $D_x \rightarrow \overline{L_y}$ denotes that $D_x$ is not written in $L_y$. For each language $L_i \in L$, each document can fall into four possible categories:

**True Positive (TP)** $D_x \rightarrow L_i$ and $D_x \triangleright L_i$

**False Positive (FP)** $D_x \rightarrow \overline{L_i}$ and $D_x \triangleright L_i$

**False Negative (FN)** $D_x \rightarrow L_i$ and $D_x \triangleright \overline{L_i}$

**True Negative (TN)** $D_x \not\rightarrow \overline{L_i}$ and $D_x \not\triangleright \overline{L_i}$

Given a gold-standard and a set of predictions, the frequency of each category can be tabulated for each language. On the basis of these counts, precision ($\mathcal{P}$) and recall ($\mathcal{R}$) are defined as the following ratio of counts:

$$\mathcal{P} = \frac{TP}{TP + FP} \qquad \mathcal{R} = \frac{TP}{TP + FN}$$

Earlier work has tended to only provide a breakdown based on the correct label (i.e. only reporting per-language recall). This gives us a sense of how likely a document in any given language is to be classified correctly, but does not give an indication of how likely a prediction for a given language is of being correct. Under the monolingual assumption (i.e. each document is written in exactly 1 language), this is not too much of a problem, as any false negative for one language must also be a false positive for another language, so precision and recall are closely linked. Nonetheless, later authors have tended to explicitly state both precision and recall for clarity. It is also common practice to report an F-score ($\mathcal{F}$), which is the harmonic mean of precision and recall:

$$\mathcal{F} = 2 \cdot \frac{\mathcal{P} \cdot \mathcal{R}}{\mathcal{P} + \mathcal{R}}$$

The F-score (also sometimes called F-measure) was developed in information retrieval to measure the effectiveness of retrieval with respect to a user who attaches different relative importance to precision and recall (van Rijsbergen 1979:Chapter 7). When used as an evaluation metric for classification tasks, it is common to place equal weight on precision and recall, and this has also been the practice in work to date on LangID that has used the F-score (Murthy and Kumar 2006; Baldwin and Lui 2010a; Takçı and Güngör 2012; Vogel and Tresner-Kirsch 2012).

In addition to evaluating performance for each individual language, authors have also sought to convey the relationship between classification errors and specific sets of languages. Errors in LangID systems are generally not random; rather, certain sets of languages are much more likely to be confused. For example, Grefenstette (1995) found that Norwegian documents had an elevated chance of being misclassified as Swedish, compared to a range of other European languages. Sibun and Reynar (1996) found an elevated chance of misclassification between Croatian, Serbian and Slovenian, and this specific set of languages has been the focus of later research (Ljubešić *et al.* 2007; Tiedemann and Ljubešić 2012). The typical method of conveying this information is through the use of a confusion matrix, a tabulation of the distribution of (predicted language, actual language) pairs. Confusion matrices can be presented over the entire language set (Sibun and Reynar 1996), or can be cropped to focus on a particular subset of languages (Kikui 1996).

Presenting full confusion matrices becomes problematic as the number of languages considered increases, and as a result has become relatively uncommon in work that covers a broader range of languages. Per-language results are also harder to interpret as the number of languages increases, and so it is common to present only collection-level summary statistics. There are two methods to summarize across a whole collection: (1) giving each document equal weight; and (2) giving each class (i.e. language) equal weight. (1) is referred to as a micro-average, and (2) as a macro-average. For LangID under the monolingual assumption, micro-averaged precision and recall are the same, since each instance of a false positive for one language must also be a false negative for another language. In other words, micro-averaged precision

and recall are both simply the collection-level accuracy. On the other hand, macro-averaged precision and recall give equal weight to each language. In datasets where the number of documents per language is the same, this again works out to being the collection-level average. However, LangID research has frequently dealt with datasets where there is a substantial skew between classes. In such cases, the collection-level accuracy is strongly biased towards more heavily-represented languages. For example, as of December 2013, English Wikipedia accounts for 4.5M of 30.5M articles (15%), whereas Chinese Wikipedia only accounts for 734k articles (2.4%).[2] Reporting the average on a representative sample of Wikipedia would mean that the English accuracy would have over 6 times the impact on the overall accuracy of the Chinese accuracy. This effect can be seen in the results reported by Baldwin and Lui (2010a), where the best performing system on a Wikipedia-based dataset has a collection-level accuracy of 86.9%, whereas the macro-averaged precision and recall are 74.0% and 64.6% respectively. To address this issue, in work on skewed document collections, authors tend to report both the collection-level accuracy as well as the macro-averaged precision/recall/F-score, in order to give a more complete picture of the characteristics of the method being studied.

Whereas the notions of macro-averaged precision and recall are clearly defined, there are two possible methods to calculate the macro-averaged F-score. The first is to calculate it as the harmonic mean of the macro-averaged precision and recall, and the second is to calculate it as the arithmetic mean of the per-class F-score. Baldwin and Lui (2010a) choose to use the second definition, which has the peculiar

[2] `http://stats.wikimedia.org/EN/TablesArticlesTotal.htm`

characteristic that the F-score may not fall between the precision and recall values.

Goldszmidt *et al.* (2013) took a different approach in dealing with the skew in available data between languages, proposing instead *weighted accuracy*, where each language is weighted using the standard deviation of the estimate of the accuracy. The number of correctly-labeled documents $m$ is modeled as a random variable, with a Binomial distribution $\sim Bin(n, p)$, where $n$ is the gold-standard number of documents for the language and $p$ is the maximum likelihood estimate of the accuracy (i.e. $p = \frac{m}{n}$). The weight $w$ for the language is defined as $w = \sqrt{\frac{n}{p(1-p)}}$, and the overall weighted accuracy of an identifier on a document collection is given by $\frac{\sum_{l \in L} w_l \cdot p_l}{\sum_{l \in L} w_l}$. This method has the advantage of allowing direct evaluation of the statistical significance of a difference between classifiers by means of a Wald test. However, its relationship with commonly-used metrics has not been sufficiently explored to enable the comparison of weighted accuracy results to macro-averaged precision/recall/F-score.

The comparability of published results is also limited by the variation in size and source of the data used for evaluation. In work to date, authors have used data from a variety of different sources to evaluate the performance of proposed solutions. Typically, data for a number of languages is collected from a single source (for example web pages (Dueire Lins and Gonçalves 2004), or a newswire source (Takçı and Güngör 2012)), and the number of languages considered varies widely (see Table 2.2). Earlier work tended to focus on a smaller number of Western European languages. Notable exceptions are Sibun and Spitz (1994), which considered 23 languages including some African and Asian languages, and Kikui (1996), which included Japanese. Later work has shifted the focus to supporting larger num-

bers of languages simultaneously, with the work of Brown (2014) pushing the upper bound, reporting a language identifier that supports over 1300 languages. The increased size of the language set considered is partly due to the increased availability of language-labeled documents from novel sources such as Wikipedia and Twitter, supplementing existing data from translations of the Universal Declaration of Human Rights, Bible translations, as well as parallel texts from MT datasets such as OPUS (Tiedemann 2012) and SETimes[3], and European Government data such as JRC-Acquis (Steinberger *et al.* 2006). This has led to a shift away from proprietary datasets such as the ECI multilingual corpus (Armstrong-Warwick *et al.* 1994) that were more commonly used in earlier research. As more languages are considered simultaneously, the accuracy of LangID systems decreases (Padró and Padró 2004; Baldwin and Lui 2010a). A particularly striking illustration of this are results for the method of Cavnar and Trenkle (1994). The original paper reports an accuracy of 99.8% over 8 European languages. Lui and Baldwin (2011) report an accuracy of 68.6% over a dataset of 67 languages from Wikipedia, and Xia *et al.* (2009) report an accuracy as low as 1.66% by applying the method to LangID of Interlinear Glossed Text (IGT) in 638 languages.

In contrast to the trend towards considering more and more languages concurrently, there has also been a tendency towards focusing on particular groups of languages that are difficult to disambiguate (Murthy and Kumar 2006; Botha and Barnard 2012; Tiedemann and Ljubešić 2012; Zampieri *et al.* 2012a; Zubiaga *et al.* 2014). This type of research tends to focus on the reasons for affinity between the

[3] `http://nlp.ffzg.hr/resources/corpora/setimes/`

languages and the particular distinguishing characteristics of each, and so reports detailed results and analysis on a small number of languages. LangID within groups of closely-related languages is an open problem, which we discuss in greater detail in Section 2.5.6.

Separate to the question of the number and variety of languages included are issues regarding the quantity of training data used. A number of studies have examined the relationship between LangID accuracy and quantity of training data through the use of learning curves (column "LC" of Table 2.2). One slight variation on this theme is Souter *et al.* (1994), which examined the quantity of training data required before no new features were observed for a language, features being character 2-grams or 3-grams. Another variation is Baldwin and Lui (2010a), which presented a breakdown of per-language accuracy according to the training data available for each language in the dataset used. The general finding is that LangID accuracy increases with more training data, though there are some authors that report an optimal amount of training data, where adding more training data decreases accuracy thereafter (Combrinck and Botha 1995; Ljubešić *et al.* 2007). Overall, it is not clear that there is a universal quantity of data that is "enough" for any language, rather this amount appears to be affected by the particular set of languages as well as the domain of the data. As a rough gauge, training data quantities on the order of 50KB per language onwards appear to be quite common (Souter *et al.* 1994; Adams and Resnik 1997; Prager 1999a; Xafopoulos *et al.* 2004; Takçı and Güngör 2012), and most of the reported learning curves appear to plateau somewhere around 100KB to 1MB per language. The breakdown presented by Baldwin and Lui (2010a) shows that with

less than 100KB per language, there are some languages where classification accuracy is near perfect, whereas there are others where it is very poor.

Another aspect that is frequently reported on is how long a sample of text needs to be before its language can be correctly detected. Column "TS" in Table 2.2 identifies papers that report results broken down by the size of the test sample. Unsurprisingly, the general consensus is that longer samples are easier to classify correctly. In most cases, the point of inflection appears to be around 100 bytes. Text samples below 100 bytes are typically reported to have elevated error rates, and the accuracy increases quickly with increasing sample size. Beyond 100 bytes, the accuracy generally tends to plateau. As with quantity of training data, this value varies with the exact set of languages considered, as well as the source of the data. There is a strong interest in classifying short segments of text, as certain applications naturally involve short text documents, such as LangID of microblog messages or search engine queries. Another area where LangID of texts as short as one word has been investigated is in the context of dealing with documents that contain text in more than one language, where word-level LangID has been proposed as a possible solution (see Section 2.5.4). These outstanding issues have led to research focused specifically on LangID of shorter segments of text, which we discuss in more detail in Section 2.5.5.

From a practical perspective, knowing the rate at which a LangID system can process and classify documents is useful as it allows a practitioner to predict the time required to process a document collection given certain computational resources. However, so many factors influence the rate at which documents are processed that comparison of absolute values across publications is largely meaningless. Instead,

it is more valuable to consider publications that compare multiple systems under controlled conditions (same computer hardware, same evaluation data, etc.). Such papers are identified in column "S" of Table 2.2. The most common observations are that classification times between different algorithms can differ by orders of magnitude (Poutsma 2002; Lui and Baldwin 2011; Lui and Baldwin 2012; Majliš 2012; Takçi and Ekinci 2012; Takçı and Güngör 2012; Brown 2013) and that the fastest methods are not always the most accurate (Poutsma 2002; Majliš 2012; Brown 2013). Beyond that, the diversity of systems tested and the variety in the test data make it hard to draw further conclusions about the relative speed of algorithms.

Where feature selection is used, the number of features retained is a parameter of interest, as it affects both the memory requirements of the LangID system as well as its classification rate. In general, a smaller feature set results in a faster and more lightweight identifier. Relatively few authors give specific details of the relationship between the number of features selected and accuracy (column "NF" of Table 2.2). This may be because in most cases, the improvement in accuracy simply plateaus with increasing feature count (Vatanen *et al.* 2010; Lui and Baldwin 2011; Brown 2012; Lui and Baldwin 2012), though the exact number of features required varies substantially with the method and the data used. At the lower end of the scale, Cavnar and Trenkle (1994) report that 300–400 features per language is sufficient, whereas at the other end, Brown (2012) finds that for the best method tested, the error rate continues to decrease up to around 5000–8000 features per language.

## 2.3 Applications

Research in LangID has cited a variety of motivations for investigating the task. In this section, we briefly summarize what these motivations are, and how their specific needs differ.

The oldest motivation for automatic LangID is perhaps in conjunction with translation (Beesley 1988; Combrinck and Botha 1995). Automatic LangID is used as a pre-processing step to determine what translation system to apply to an input text, whether it be by routing to a specific human translator or by applying MT. Such a use case is still very common, and can be seen in the Google Chrome web browser,[4] where an in-built LangID module is used to offer MT services to the user when the detected language of the web page being visited differs from the user's language settings.

NLP components such as part-of-speech (POS) taggers and parsers tend to make a strong assumption that the input text is monolingual in a given language. Similarly to the translation case, LangID can play an obvious role in routing documents written in different languages to NLP components tailored to those languages. More subtle is the case of documents with mixed multilingual content, the most commonly-occurring instance of which is foreign inclusion, where a document is predominantly in a single language (e.g. German or Japanese) but is interspersed with words and phrases (often technical terms) from a language such as English. For example, Alex *et al.* (2007) found that around 6% of word tokens in German text sourced from the Internet are English inclusions. In the context of POS tagging, one strategy for dealing with inclusions is to have a dedicated POS for all foreign words, and force the POS tagger

[4] `http://www.google.com/chrome`

to perform both foreign inclusion detection and POS tagging of those words in the target language; this is the approach taken in the Penn POS tagset, for example (Marcus *et al.* 1993). An alternative strategy is to have an explicit foreign inclusion detection pre-processor (e.g. based on monolingual lexicon and corpus analysis: Alex (2006)), and some special handling of foreign inclusions. For example, in the context of German parsing, Alex *et al.* (2007) used foreign inclusion predictions to restrict the set of (German) POS tags used to form a parse tree, and found that this approach substantially improved parser accuracy.

Another commonly-mentioned use case is for multilingual document storage and retrieval. A document retrieval system (such as, but not limited to, a web search engine) may be required to index documents in multiple languages. In such a setting, it is common to apply LangID at two points: (1) to the documents being indexed; and (2) to the queries being executed on the collection. Simple keyword matching techniques can be problematic in text-based document retrieval, because the same word can be valid in multiple languages. Classic examples of such words (known as "false friends") include *gift*, which in German means "poison", and *burro*, which means "butter" in Italian but "donkey" in Spanish. Performing LangID on both the document and the query helps to avoid confusion between such terms, by taking advantage of the context in which it appears in order to infer the language. This has resulted in specific work in LangID of web pages (Kikui 1996; Xafopoulos *et al.* 2004; Martins and Silva 2005; Rehurek and Kolkus 2009; Chew *et al.* 2011), as well as search engine queries (Ceylan and Kim 2009; Gottron and Lipka 2010). Having said this, in many cases, the search query itself does a sufficiently good job of select-

ing documents in a particular language, and overt LangID is often not performed in mixed multilingual search contexts. Indeed, outside of specialized tasks such as cross-lingual information retrieval (Grefenstette 1998; Nie 2010), there is very little research that has shown that LangID empirically boosts retrieval effectiveness, with Han *et al.* (2011) being a rare instance of this, in demonstrating for monolingual English information retrieval over Twitter, that explicit LangID to filter out non-English documents improved retrieval effectiveness.

Automatic LangID has also been used to facilitate linguistic and other text-based research. Suzuki *et al.* (2002) report that their motivation for developing a language identifier was "to find out how many web pages are written in a particular language". Automatic LangID has been used in constructing web-based corpora. The Crúbadán project (Scannell 2007) makes used of automated LangID techniques to gather linguistic resources for under-resourced languages. Similarly, the Online Database of INterlinear text (ODIN) (Lewis and Xia 2010) uses automated LangID as one of the steps in collecting interlinear glossed text from the web for purposes of linguistic search and bootstrapping NLP tools. hrWac (Ljubešić and Klubička 2014) is a web corpus collected from the `.hr` top-level domain using automated LangID techniques (Stupar *et al.* 2011). ClueWeb09[5] is a multilingual dataset that presents web pages grouped into 10 languages; the grouping was carried out using an implementation of the method of Cavnar and Trenkle (1994). Resnik (1999) demonstrated that LangID can be used to reduce false positives in parallel corpora crawled from the web based on structural parallelism. Finally, Kralisch and Mandl (2006) make use of automated

[5] `http://lemurproject.org/clueweb09/`

language identification in the study of the effect of language on information access in websites.

One challenge in collecting linguistic resources from the web is that documents can be multilingual (i.e. contain text in more than one language). This is problematic for standard LangID methods, which assume that a document is written in a single language, and has prompted research into segmenting text by language (Yamaguchi and Tanaka-Ishii 2012), as well as word-level LangID (King and Abney 2013; Nguyen and Dogruoz 2013), to enable extraction of linguistic resources from multilingual documents.

Automated LangID also has a role in online communities such as web forums and social media. Many such communities are multilingual (e.g. Nguyen and Dogruoz (2013) report experiments on a mixed Dutch-Turkish web forum). From a usability perspective, LangID is needed to help users understand content generated by speakers of a different language (Kralisch and Mandl 2006). Such problems are often faced by users of travel-related websites such as TripAdvisor[6] or Booking.com,[7] where other users may have written reviews in a language we are not familiar with. Furthermore, when scraping websites containing user-generated content, it is common to encounter web pages that contain text in more than one language (an issue discussed in Section 2.5.4). Another common issue is that the language of the content may not match the language of the interface used (Bosca and Dini 2010). Amongst the newer "social media" platforms, a number of authors have investigated LangID for Twitter messages (Tromp and Pechenizkiy 2011; Bergsma *et al.* 2012; Lui and Baldwin 2012;

[6] `http://www.tripadvisor.com`

[7] `http://booking.com`

Mayer 2012; Carter *et al.* 2013; Derczynski *et al.* 2013; Goldszmidt *et al.* 2013; Lui and Baldwin 2014), and Han *et al.* (2014a) demonstrated that when performing user geolocation over Twitter, it is empirically advantageous to first automatically partition users based on language, based on the observation that for many languages, the mere fact that a given user posts in the language biases the set of locations they are likely to be based in (e.g. if a user posts in Finnish, they are highly likely to be based in Finland, and not elsewhere in the world). There has also been research on identifying the language of private messages between eBay users (Mayer 2012), presumably as a filtering step prior to more in-depth data analysis.

## 2.4 Off-the-Shelf Language Identifiers

In Section 2.1, we attributed part of the popularity of the method of Cavnar and Trenkle (1994) to the availability of `TextCat`, an "off-the-shelf" implementation of the algorithm. In this context, "off-the-shelf" implies that the software is distributed with pre-trained models for a number of languages, such that a user is not required to provide training data before using the system. Such a setup is highly attractive to many end-users of automatic LangID, whose main interest is in utilizing the output of a language identifier rather than implementing and developing the technique. To this end, a number of off-the-shelf language identifiers have been released over time. In this section, we provide a brief summary of systems that are available, as well as the key characteristics of each system.

`TextCat` is the most well-known implementation of Cavnar and Trenkle (1994), and is still available (van Noord 1994), and as of January 2014 lists models for 76

languages in its off-the-shelf configuration. The classifier itself uses rank-order statistics of byte $n$-grams to determine the most likely language for a document, and is described in detail in Section 4.1.1. `TextCat` is not the only example of an off-the-shelf implementation of Cavnar and Trenkle (1994); another such implementation is Scheelen (2003). The main issue addressed by later implementations is classification speed: `TextCat` is implemented in Perl and is not optimized for speed, whereas implementations such as Scheelen (2003) have been specifically written to be fast and efficient.

`ChromeCLD` (Sites 2013b) is the language identifier embedded in the Google Chrome web browser.[8] It uses a naive Bayes classifier, and script-specific classification strategies. `ChromeCLD` assumes that all input is in UTF-8, and assigns the responsibility of encoding detection and transcoding to the user. `ChromeCLD` uses Unicode information to determine the script of the input, which is then processed in one of three ways: (1) if the script is specific to a single language, that language is immediately output as a prediction; (2) if the input contains Chinese/Japanese/Korean scripts, a character-level unigram model is used; and (3) all other documents are classifier using a character-level quadgram (4-letter) model. `ChromeCLD` also implements a number of pre-processing heuristics to help boost performance on its target domain (web pages), such as stripping letter sequences like '.jpg'. The standard implementation supports 83 languages, and an extended model is provided that supports 160 languages.

`LangDetect` (Nakatani 2010b) is a Java library that implements a language identifier based on a naive Bayes classifier trained over character $n$-grams. As of Jan-

[8] `http://www.google.com/chrome`

uary 2014, the software comes with pre-trained models for 53 languages, using data from Wikipedia. `LangDetect` makes use of a range of normalization heuristics to improve the performance on particular languages. These include: (1) clustering of Chinese/Japanese/Korean characters to reduce sparseness; (2) removal of "language-independent" characters, and other text normalization; and (3) normalization of Arabic characters. We examine `LangDetect` in greater detail in Section 4.1.3.

`whatlang` (Brown 2013) uses a vector-space model with per-feature weighting on character $n$-gram sequences. It is a standalone version of an identifier originally developed to produce a "language-aware" version of the Unix `strings` utility (Brown 2012). One particular feature of `whatlang` is that it uses discriminative training in selecting features, i.e. it specifically makes use of features that are strong evidence *against* a particular language, which is generally not done by naive Bayes models, and has been shown to be effective, particularly in discriminating between closely-related languages (Tiedemann and Ljubešić 2012). Another feature of `whatlang` is that it uses inter-string smoothing to exploit sentence-level locality in making language predictions, under the assumption that adjacent sentences are likely to be in the same language. Brown (2013) reports that this substantially improves the accuracy of the identifier. Another distinguishing feature of `whatlang` is that it comes pre-trained with data for 1100 languages, which is the highest number by a large margin of any off-the-shelf system.

`YALI` (Majliš 2012) implements an off-the-shelf classifier trained using Wikipedia data, covering 122 languages. Although not described as such, the actual classification algorithm used is a linear model, and is thus closely related to both multinomial

naive Bayes and a cosine-based vector space model (see Section 5.5.1). The weights for each term are calculated as the maximum likelihood estimate. However, unlike in a multinomial naive Bayes model, the weights are not log-scaled. The feature representation used is byte-level 4-grams, selecting the most frequent 100 per-language.

In addition to the above-mentioned general-purpose language identifiers, there have also been recent efforts to produce pre-trained language identifiers targeted specifically at Twitter messages. `LDIG` (Nakatani 2012) is a Twitter-specific LangID tool with in-built models for 17 languages. It uses a document representation based on tries (Okanohara and Tsujii 2009). The algorithm is a logistic regression classifier using all possible substrings of the data. The use of all possible substrings is important to maximize the available information from the relatively short Twitter messages. Okanohara and Tsujii (2009) show how to construct a maximal substring model that is equivalent to the all substring model, with the advantage that the maximal model can be constructed in linear time, and utilizes tries for fast prediction. Nakatani (2012) also applies some hand-crafted normalization rules to further boost accuracy, similar to those applied in earlier work by the same author (Nakatani 2010b).

Another Twitter-specific LangID tool is `MSR-LID` (Goldszmidt *et al.* 2013) which, like `TextCat`, uses rank-order statistics over character $n$-grams. Unlike `TextCat`, an unnormalized variant of Spearman's $\rho$ is used to measure correlation. The Twitter-specific training data is acquired through a bootstrapping approach, starting with Wikipedia data. The authors provide a reference implementation, as well as pre-trained models for a variety of parametrizations of the system.

## 2.5 Open Issues in LangID

As we have seen in the previous sections, there is wide diversity in the types of research that are relevant to LangID. In this section, we identify specific issues or research questions that have been raised over the years with regards to particular aspects of the LangID problem, and provide a short analysis of work to date on each issue.

Several papers have been published cataloging open issues in LangID (Sibun and Reynar 1996; Xia *et al.* 2010b; Hughes *et al.* 2006; Da Silva and Lopes 2006; Baldwin and Lui 2010a). Some of the issues, such as text representation (Section 2.2.1, Section 2.2.2) and choice of algorithm (Section 2.2.3), have already been covered in detail in this review. In this section, we synthesize the remaining issues into a single master list, adding our own where appropriate. The following subsections each explore an issue, and work to date that has been done to address it.

### 2.5.1 Text Preprocessing

Text preprocessing (also known as normalization) is an umbrella term for techniques where an automatic transformation is applied to text before it is presented to a classifier. The aim of such a process is to eliminate sources of variation that are expected to be confounding factors with respect to the target task. Text preprocessing is slightly different from data cleaning, as data cleaning is a transformation applied only to training data, whereas normalization is applied to both training and test data. Hughes *et al.* (2006) raise text preprocessing as an outstanding issue in LangID, arguing that its effects on the task have not been sufficiently investigated.

While there is no work to date that specifically focuses on comparing different approaches to text preprocessing for the explicit purposes of LangID, in this section we summarize the normalizations that have been proposed for LangID.

Case folding is the elimination of capitalized letters, replacing them with their lowercased forms. This is not a trivial operation, as it requires script-specific information, and so is generally only possible if the exact encoding is known. This can be problematic when dealing with encodings that share certain portions of the codespace, such as many 8-bit encodings which share the lower 7-bit space with ASCII. In ASCII, uppercase letters can be converted to lowercase by adding 32 to the value of each byte, and this rule extends to some members of the ISO-8859 family, but encodings such as CP850 do not follow this rule, potentially resulting in incomplete (if only the ASCII portion is converted) or incorrect (if the add-32 rule is blindly applied) case folding. One solution to this problem is provided by the Unicode Character Database (UCD),[9] which explicitly represents case relationships between symbols, making automatic case folding relatively straightforward if the input document is known to be in a Unicode encoding.

Goldszmidt *et al.* (2013) apply case folding in building a language identifier for Twitter messages, and find that case folding generally lowers accuracy. `ChromeCLD` (Sites 2013b) makes use of case folding, but does not report on its effect on accuracy. `ChromeCLD` also makes use of a variety of other heuristics, including expanding HTML entities, deleting digits and punctuation, and removing SGML-like tags. `LangDetect` (Nakatani 2010a) is another off-the-shelf language identifier that makes some use

[9] `http://www.unicode.org/ucd/`

of normalization techniques. Characters from Chinese/Japanese/Korean scripts are clustered together to reduce their relative sparsity – script information is again obtained from UCD. `LangDetect` also removes "language-independent characters" such as numbers, symbols, URLs and mail addresses. It also removes words that are all-capitals, and tries to remove other acronyms and proper names, though how this is done is not specified (Nakatani 2010b).

In the domain of Twitter messages, Tromp and Pechenizkiy (2011) remove links, usernames, smilies and hashtags (a Twitter-specific "tagging" feature), arguing that these entities are language independent and thus should not feature in the model. However, they do not report on the effect that this cleaning has on accuracy. Xafopoulos *et al.* (2004) address LangID of web pages, and report removing HTML formatting, and applying stopping using a small stopword list. They also do not report the effect that this has on accuracy. Takçi and Ekinci (2012) carry out LangID experiments on the ECI multilingual corpus, and report removing punctuation, space characters and digits. They too do not report the effect of the cleaning on accuracy.

### 2.5.2 Supporting Lower-Density Languages

Hughes *et al.* (2006) paint a fairly bleak picture of the support for lower-density languages in automatic LangID, and this is supported by the arguments of Xia *et al.* (2010b), who detail specific issues in building hugely multilingual datasets. Abney and Bird (2010) has also specifically called for research into automatic LangID for low-density languages. As we saw in Table 2.2, early research in LangID tended to focus on a very limited number of languages (sometimes as few as 2). This situ-

Rhoddodd yr athro lyfr i'r bachgen ddoe
gave-3sg the teacher book to-the boy yesterday
'The teacher gave a book to the boy yesterday'

Figure 2.2: Example of Interlinear Glossed Text (IGT), reproduced from Xia *et al.* (2010a).

ation has improved somewhat, with many current off-the-shelf language identifiers supporting on the order of 50-100 languages (Section 2.4). The standout in this regard is Brown (2013), supporting 1100 languages in its default configuration. This is still substantially less than the 7100 languages listed by the Ethnologue (Lewis *et al.* 2014). However, only about half the listed languages are known to have a writing system, and in many cases the writing system is very rarely used, with little to no digitized text available.[10] Xia *et al.* (2010b) describe the Ethnologue in more detail, and discuss the role that LangID plays in other aspects of supporting minority languages, including detecting and cataloging resources. The problem is circular: LangID methods are typically supervised, and need training data for each language to be covered, but the most efficient way to recover such data is through LangID methods. Brown (2013) avoids this issue by using data from the most varied source available (and part of the *raison d'etere* of Ethonologue): bible translations. However, as we will see in Chapter 4, bible text tends to be regular in a way that is often not representative of other forms of text in a language.

A number of projects are ongoing with the specific aim of gathering linguistic data from the web, targeting as broad a set of languages as possible. One such project is the Online Database of INterlinear text (ODIN) (Xia *et al.* 2010a), which

[10] `http://www.ethnologue.com/enterprise-faq/how-many-languages-world-are-unwritten`

aims to collect parallel snippets of text from scholarly articles published to the web. ODIN specifically targets articles containing Interlinear Glossed Text (IGT), a semi-structured format for presenting text and a corresponding gloss that is commonly used in linguistics. Figure 2.2 reproduces an example of IGT reported in Xia *et al.* (2010a). Xia *et al.* (2010a) describe the construction of web crawlers specifically targeting IGT, as well as the identification of the languages represented in the IGT snippets. Language identification for thousands of languages from very small quantities of text is one of the issues that they have had to tackle. They list four specific challenges for LangID in ODIN: (1) the large number of languages, (2) "unseen" languages, that appear in the test data but not in training data, (3) short target sentences, and (4) (sometimes inconsistent) transliteration into Latin text. They report that the method of Cavnar and Trenkle (1994) attains an accuracy as low as 1.66% on their specific task. Their solution to this task is to take advantage of a domain-specific feature: they assume that the name of the language that they are extracting must appear in the document containing the IGT. The first step is IGT and language mention detection, which is carried out in a language-independent fashion. After language names and IGT have been identified, the next step is to map each extent of IGT to a language name, and this is done using a machine learning approach similar to those used for coreference resolution, using features such as the nearest language that precedes the IGT instance, language names appearing in the neighbourhood of the IGT instance, as well as supervised classification and unsupervised clustering of IGT instances using a character and word level $n$-gram representation. Xia *et al.* (2010a) report that this approach significantly outperforms the text-based LangID approach

in this particular problem setting.

Another project with the aim of creating text corpora for under-resourced languages by crawling the web is the Crúbadán project (Scannell 2007). The crawler uses seed data in a target language to generate word lists, that in turn are used as queries for a search engine. The returned documents are then compared to the seed resource via an automatic language identifier, which is used to eliminate false positives. The core identifier uses a vector-space model with manually-set minimum thresholds, and is augmented in problematic cases with a word-level naive Bayes classifier. Scannell (2007) reports that corpora for over 400 languages have been built using this method.

Much recent work on multilingual documents (Section 2.5.4) has been done with support for minority languages as a key goal. One of the common problems with gathering linguistic data from the web is that the data in the target language is often embedded in a document containing data in another language. This has spurred recent developments in text segmentation by language (Yamaguchi and Tanaka-Ishii 2012) and word-level LangID (King and Abney 2013; Nguyen and Dogruoz 2013), which we discuss in more detail in the section on LangID for multilingual documents (Section 2.5.4).

### 2.5.3 "Unseen" Languages

"Unseen" languages are languages that we do not have training data for, that may nonetheless be encountered by a language identifier system being applied to real-world data. Dealing with languages for which we do not have training data has

been identified as an issue by Hughes *et al.* (2006), and has also been mentioned by Xia *et al.* (2010a) as a specific challenge in harvesting linguistic data from the web. Xia *et al.* (2010a) address the problem by using a specific feature of the type of data they are targeting: they assume that the name of the language appears somewhere in the document they are extracting strings from, and thus tackle the issue through coreference resolution, which we describe in more detail in Section 2.5.2. This is an elegant solution to their problem but it is not applicable to general text-based LangID.

Some authors have attempted to tackle the unseen language problem through attempts at unsupervised labeling of text by language. Biemann and Teresniak (2005) present such an approach, building graphs of co-occurrences of words in sentences, and then partitioning the graph using a custom graph-clustering algorithm, which labels each word in the cluster with a single label. The number of labels starts out the same as the number of words, and decreases as part of the process of applying the algorithm. After a small number of iterations (the authors report 20), the labels become relatively stable and can be interpreted as cluster labels. Smaller clusters are then discarded, and the remaining clusters are interpreted as groups of words for each language. The authors apply this algorithm to a corpus containing documents in 7 European languages, and report that (after manually mapping clusters to languages) the LangID accuracy is comparable to supervised methods.

Amine *et al.* (2010) also tackle the unseen language problem through clustering. In contrast to the word co-occurrence statistics used by Biemann and Teresniak (2005), they use a character $n$-gram representation for text, and a clustering algorithm that

consists of an initial K-means phase, followed by a particle-swarm optimization designed to mimic the behavior of ants (Deneubourg *et al.* 1990). This produces a large number of small clusters, which are then labeled by language through a separate step.

Biemann and Teresniak (2005) and Amine *et al.* (2010) both tackle the unseen languages problem by treating it as an unsupervised learning (i.e. clustering) problem. However, both solutions are somewhat unsatisfactory as there is still a step required to label the clusters, so essentially the problem has simply been transformed from document-level supervised LangID to cluster-level supervised LangID with manual intervention. The fact that clustering algorithms can group documents from related languages is promising, but there is much work to be done in this respect in terms of actually developing techniques that are useful for extracting text samples from resources such as the web for languages that we do not have training data for.

A different incomplete solution to the issue of unseen languages is to design the classifier to be able to output "unknown" as a prediction for language (Biemann and Teresniak 2005; Rehurek and Kolkus 2009; Winnemöller 2010). This helps to alleviate one of the problems commonly associated with the presence of unseen languages – classifiers without an "unknown" facility are forced to pick a language for each document, and in the case of unseen languages the choice may be arbitrary and unpredictable (Biemann and Teresniak 2005). When LangID is used for filtering purposes, i.e. to select documents in a single language, this mislabeling can introduce substantial noise into the data extracted; furthermore, it does not matter what or how many unseen languages there are, as long as they are consistently rejected. Therefore the "unknown" output provides an adequate solution to the unseen language problem

for purposes of filtering.

The easiest method to implement unknown language detection is through thresholding. Most systems internally compute a score for each language for an unknown text, so thresholding can be applied either with a global threshold (Cowie *et al.* 1999), a per-language threshold, or by comparing the score for the top-scoring N-languages. Of the off-the-shelf systems we discussed in Section 2.4, only `ChromeCLD` implements unknown language detection. However, there has been relatively little attention paid to this issue, and there is a lack of research into how to evaluate such "unknown" predictions (e.g. should a system be equally penalized for wrongly predicting "unknown" as for predicting the wrong language?).

### 2.5.4 Multilingual Documents

Multilingual documents are documents that contain text in more than one language. Recent research has investigated how to make use of multilingual documents from sources such as web crawls, (King and Abney 2013), forum posts (Nguyen and Dogruoz 2013) and microblog messages (Ling *et al.* 2013). However, most LangID methods assume that a document contains text from a single language, and so are not directly applicable to multilingual documents. As a result, research to date has sometimes discarded multilingual documents before carrying out experiments (Cavnar and Trenkle 1994; Tromp and Pechenizkiy 2011).

Handling of multilingual documents has been named as an open research question (Hughes *et al.* 2006). Most natural language processing techniques presuppose monolingual input data, so inclusion of data in foreign languages introduces

noise, and can degrade the performance of NLP systems (Alex *et al.* 2007; Cook and Lui 2012). Automatic detection of multilingual documents can be used as a pre-filtering step to improve the quality of input data. Detecting multilingual documents is also important for acquiring linguistic data from the web (Scannell 2007; Abney and Bird 2010), and has applications in mining bilingual texts for statistical machine translation from online resources (Resnik 1999; Nie *et al.* 1999; Ling *et al.* 2013), or to study code-switching phenomena in online communications (Nguyen and Dogruoz 2013). There has also been interest in extracting text resources for low-density languages from multilingual web pages containing both the low-density language and another language such as English (Yamaguchi and Tanaka-Ishii 2012; King and Abney 2013). King and Abney (2013:p1118) specifically mention the need for an automatic method "to examine a multilingual document, and with high accuracy, list the languages that are present in the document".

The need to handle multilingual documents has prompted researchers to revisit the granularity of LangID. Many researchers consider document-level LangID to be relatively easy (McNamee 2005), and that sentence-level (Brown 2013) and word-level (King and Abney 2013; Nguyen and Dogruoz 2013) LangID are more suitable targets for further research. However, word-level and sentence-level tokenization are not language-independent tasks, and for some languages are substantially harder than others (Peng *et al.* 2004). Furthermore, reducing the granularity of LangID also presents challenges in dealing with shorter quantities of text on which to base the prediction (see Section 2.5.5).

Research to date on LangID for multilingual documents has been fairly limited.

`Linguini` (Prager 1999a) is a language identifier that supports identification of multilingual documents, which we analyze in greater detail in Chapter 4. The system is based on a vector space model, and cosine similarity between a feature vector for the test document and a feature vector for each language $L_i$, computed as the sum of feature vectors for all the documents for language $L_i$ in the training data. The elements in the feature vectors are frequency counts over byte $n$-grams ($2 \leq n \leq 5$) and words. Language identification for multilingual documents is performed through the use of *virtual mixed languages*. Prager (1999a) shows how to construct vectors representative of particular combinations of languages independent of the relative proportions, and proposes a method for choosing combinations of languages to consider for any given document. One weakness of this approach is that for exhaustive coverage, this method is factorial in the number of languages, and as such intractable for a large set of languages. Furthermore, calculating the parameters for the virtual mixed languages becomes unfeasibly complex for mixtures of more than 3 languages (see Section 6.2.4).

Another approach to handling multilingual documents is to attempt to segment them into contiguous monolingual segments. In addition to identifying the languages present, this requires identifying the locations of boundaries in the text which mark the transition from one language to another. Several methods for supervised language segmentation have been proposed. Teahan (2000) proposed a system based on text compression that identifies multilingual documents by first segmenting the text into monolingual blocks. Mandl *et al.* (2006) detect "language shift" using an eight-word sliding window. Rehurek and Kolkus (2009) perform language segmentation by

computing a relevance score between terms and languages, smoothing across adjoining terms and finally identifying points of transition between high and low relevance, which are interpreted as boundaries between languages. Yamaguchi and Tanaka-Ishii (2012) use a minimum description length approach, embedding a compressive model to compute the description length of text segments in each language. They present a linear-time dynamic programming solution to optimize the location of segment boundaries and language labels.

Closely related to the idea of text segmentation by language is the idea of word-level LangID (King and Abney 2013; Nguyen and Dogruoz 2013). Here, the task becomes to label each word in the document with a specific language. Work to date in this area has assumed that word tokenization can be carried out on the basis of whitespace, and that the languages present in the document are known in advance. King and Abney (2013) make use of conditional random fields, and introduce a technique to estimate the parameters using only monolingual data, an important consideration as there is no readily-available collection of manually-labeled multilingual documents with word-level annotations. Nguyen and Dogruoz (2013) present a two-pass approach to processing Turkish-Dutch bilingual documents, where the first pass labels each word independently and the second pass uses the local context of a word to further refine the predictions.

To encourage further research on LangID for multilingual documents, the Australiasian Language Technology Workshop 2010 hosted a shared task where participants were required to predict the language(s) present in a held-out test set containing monolingual and bilingual documents (Baldwin and Lui 2010b). The dataset was

prepared using data from Wikipedia, and bilingual documents were produced using a segment from an article in one language, and a segment from the equivalent article in another language. Equivalence between articles was determined using the cross-language links embedded within each Wikipedia article.[11] The winning entry (Tran *et al.* 2010) attained a macro-averaged F-score of 0.932, by first building monolingual models from multilingual training data, and then applying them to a chunked version of the test data and making the final prediction a function of the prediction over chunks.

### 2.5.5 Short Texts

Language identification of short strings has attracted recent interest from the research community. Hammarström (2007) describes a method that augments a dictionary with an affix table, and tested it over synthetic data derived from a parallel bible corpus. Ceylan and Kim (2009) compared a number of methods for identifying the language of search engine queries of 2 to 3 words. They develop a method which uses a decision tree to integrate outputs from several different LangID approaches. Vatanen *et al.* (2010) focus on messages of 5 – 21 characters, using $n$-gram language models over data drawn from UDHR in a naive Bayes classifier. Carter *et al.* (2013) focus specifically on LangID in Twitter messages by augmenting standard methods with LangID priors based on a user's previous messages and by the content of links embedded in messages; this is the method used in `TwitIE` (Bontcheva *et al.* 2013).

[11] Note that such articles are not necessarily direct translations but rather articles about the same topic written in different languages.

Tromp and Pechenizkiy (2011) present a method for LangID of short text messages by means of a graph structure, extending the standard 'bag' model of text to include information about the relative order of tokens. This method was further developed by Vogel and Tresner-Kirsch (2012), who proposed linguistically-motivated modifications to the algorithm of Tromp and Pechenizkiy (2011). Their proposed augmentations include the use of word-length information, as well as downweighting of repeated information and improving robustness to outliers through the use of medians rather than averages. Bergsma *et al.* (2012) examine LangID for creating language-specific twitter collections, finding that a compressive method trained with out-of-domain data from Wikipedia and standard text corpora performed better than the off-the-shelf language identifiers they tested. Goldszmidt *et al.* (2013) proposed a method based on rank-order statistics, using a bootstrapping process to acquire in-domain training data from unlabeled Twitter messages.

Whilst all of the above-mentioned approaches are similar in that they specifically tackle LangID of short text segments, there are significant differences between them due to the specific domains they target. The work of Hammarström (2007) and Vatanen *et al.* (2010) is domain-agnostic, in that their focus is on accurate LangID of short text segments. We would expect that generic methods for LangID of short texts should be effective in any domain where short texts are found, such as search engine queries or microblog messages. However, Hammarström (2007) and Vatanen *et al.* (2010) both only test their systems in a single domain, Bible texts in the former case and texts from the Universal Declaration of Human Rights (UDHR) in the latter case. Other research has shown that LangID results do not trivially generalize

across domains (Baldwin and Lui 2010a; Lui and Baldwin 2011), and found that LangID in UDHR documents is relatively easy (Yamaguchi and Tanaka-Ishii 2012). For both Bible and UDHR data, we expect that the linguistic content is relatively grammatical and well-formed, an expectation that does not carry across to domains such as search engine queries and microblogs. In the absence of further empirical evidence, it is difficult to conclude whether the proposed systems would be effective in such application domains.

In the domain of search engine queries, the method of Ceylan and Kim (2009) appears to be state-of-the-art, and draws on features derived from the work of Hammarström (2007). Overall, they report that their method significantly outperforms that of Cavnar and Trenkle (1994) (82.7% vs 65.2% accuracy).

There has been more work done in the microblog domain, with Tromp and Pechenizkiy (2011), Carter *et al.* (2013) and Goldszmidt *et al.* (2013) all reporting over 90% accuracy on Twitter messages. One significant difference in the approach of Carter *et al.* (2013) is that they make use of additional domain-specific information in the form of semi-supervised priors, drawn from metadata related to the message such as the author as well as the language of pages linked to by the message (we discuss the use of metadata to facilitate LangID in Section 2.5.8). From a practical standpoint, the text-only approaches of Tromp and Pechenizkiy (2011) and Goldszmidt *et al.* (2013) are more attractive, as they require tracking and processing of much less information. However, if the goal is to maximize accuracy at all costs, it may be possible to integrate these approaches with the semi-supervised priors of Carter *et al.* (2013), potentially leading to a better-performing hybrid system.

Bergsma *et al.* (2012) raise an important criticism of LangID work on Twitter messages to date: only a small number of European languages has been considered. Baldwin and Lui (2010a) showed that for longer documents, good performance on just European languages did not necessarily imply equally good performance in the general case, and it stands to reason that in the case of short text segments the problem is exacerbated as more languages are considered. This does not detract from the work of Tromp and Pechenizkiy (2011) and Carter *et al.* (2011), but rather highlights the need for further research. Bergsma *et al.* (2012) expand the scope of LangID for Twitter, covering nine languages across Cyrillic, Arabic and Devanagari scripts. In Chapter 7, we tackle the problem of gathering data to evaluate the accuracy of LangID on Twitter messages across a broader range of languages.

### 2.5.6 Closely-related Languages

While one line of research into LangID has focused on pushing the boundaries of how many languages are supported simultaneously by a single system (Xia *et al.* 2010b; Brown 2012; Brown 2013), another has taken a complementary path and focused on LangID in groups of closely-related languages. Research in the area typically does not make a distinction between languages, varieties and dialects, because such terminological differences tend to be political rather than linguistically-motivated (Xia *et al.* 2010b; Zampieri *et al.* 2012b), and the technical challenges presented tend to be fairly similar.

Language identification for closely-related languages has been studied for Malay-Indonesian (Ranaivo-Malancon 2006), Indian languages (Murthy and Kumar 2006),

Serbo-Croatian languages (Ljubešić *et al.* 2007; Tiedemann and Ljubešić 2012), Australian-British-Canadian English (Lui and Cook 2013), Belgian-Netherlandic Dutch (Peirsman *et al.* 2010), Dutch dialects (Trieschnigg *et al.* 2010), Mainland-Singapore-Taiwan Chinese (Huang and Lee 2008), European-Brazilian Portuguese (Zampieri *et al.* 2012b), Spanish varieties (Zampieri *et al.* 2013), French varieties (Diwersy *et al.* 2014), and Arabic dialects (Elfardy and Diab 2013; Zaidan and Callison-Burch 2014).

Closely-related languages are a known problem for existing language identifiers (Ranaivo-Malancon 2006; Sites 2013a; Zampieri 2013). Tiedemann and Ljubešić (2012) report an overall accuracy of 97.7% on Bosnian/Serbian/Croatian, compared to 45% attained by `TextCat`. Lui and Cook (2013) find that LangID methods are not competitive with word-based methods in distinguishing between national varieties of English. Ranaivo-Malancon (2006) reports that a character trigram model is able to distinguish Malay/Indonesian from English, French, German and Dutch, but handcrafted rules are needed to distinguish between Malay and Indonesian. One kind of rule is the use of "exclusive words" that are known to occur in only one of the languages. A similar idea is used by Tiedemann and Ljubešić (2012), which automatically learn a "blacklist" of words that have a strong negative correlation with a language – i.e. their presence implies that the text is *not* written in a particular language. Brown (2013) also adopts such "discriminative training" to make use of negative evidence in LangID.

Zampieri (2013) investigated the issue of document representation for closely related languages, since typical LangID approaches use a character $n$-gram representation of text, but recent work on closely-related languages seems to favor

word-based representations (Huang and Lee 2008; Tiedemann and Ljubešić 2012; Lui and Cook 2013), comparing $n$-gram based representations to bag-of-words representations for LangID over varieties of Spanish, Portuguese and French. The results were inconclusive, with word-level models being better for Spanish and character $n$-gram models being better for Portuguese and French.

The 25th International Conference on Computational Linguistics (Coling 2014) in Dublin, Ireland included a workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial). The workshop proposed a shared task on discriminating between similar languages (Zampieri *et al.* to appear), where participants were challenged to build an automatic LangID system to discriminate between 13 languages in 6 groups. Groupings included highly-similar languages, as well as national varieties of the same language. Results were generally strong, consistent with previous work showing that targeted classifiers are able to discriminate between closely-related languages with high accuracy. However, the broader problem of integrating these results into a general LangID system was not directly addressed.

### 2.5.7 Encoding, Orthography and Transliteration

The character $n$-gram models that are typical in LangID (see Section 2.2.1) model text as consisting of a stream of characters. However, there is a slight mismatch between this view and how text is actually stored: documents are digitized using a particular encoding, which is a mapping from characters (e.g. a letter in the alphabet, or a Chinese ideogram), onto the actual sequence of bytes that is stored and transmitted by computers. Well-known and commonly-used encodings include

ASCII, Latin-1 and UTF8. For many encodings there is a one-to-one mapping between bytes and the characters they represent (e.g. in ASCII, characters are always represented by exactly one byte). Where this is the case, there is no practical difference between a stream-of-bytes and a stream-of-characters representation. However, there are two complicating factors in approximating the stream of characters with an underlying stream of bytes: (1) some encodings represent certain characters using multi-byte sequences, and these sequences can be of variable length (e.g. UTF8); and (2) some languages have several common encodings in use (e.g. Chinese is often encoded in GuoBiao, Big5 and Unicode-based encodings). Some research has avoided the issue entirely by assuming that all the documents to be processed use the same encoding, which may be a reasonable assumption in some settings, such as when processing data from a single source (e.g. all data from Twitter and Wikipedia is UTF-8 encoded). `ChromeCLD` (Sites 2013b) also assumes that all input data is UTF-8 encoded. However, in formulating a generalized language identifier, assuming a fixed encoding simply defers the problem. Some research has included an encoding detection step to resolve bytes to the characters they represent (Kikui 1996), effectively transcoding the document into a standardized encoding before attempting to identify the language. However, transcoding is computationally expensive, and other research suggests that it may be possible to ignore encoding and build a single per-language model covering multiple encodings simultaneously (Kruengkrai *et al.* 2005; Baldwin and Lui 2010a). Another solution is to treat each language-encoding pair as a separate category (Cowie *et al.* 1999; Suzuki *et al.* 2002). The disadvantage of this is that it increases the sparsity of the training data, and increases computational

costs by modeling a larger number of classes.

Related to issues of encoding are issues of orthography and transliteration. Certain languages can be written in more than one script, for example Serbian, which is commonly written using both Latin and Cyrillic script. Like encoding, there is generally a one-to-one mapping between different orthographies for the same language, and so a similar issue arises that what is logically the same text can have multiple concrete representations. Transliteration is another phenomenon that has a similar effect, whereby phonetic transcriptions in another script are produced for particular languages. These transcriptions can either be standardized and officially sanctioned, such as the use of *Hanyu Pinyin* for Chinese, or may also emerge irregularly and organically as in the case of *arabizi* for Arabic (Yaghan 2008). Correctly and automatically identifying instances of transcribed text is an open question for LangID research.

### 2.5.8 Domain-specific LangID

One approach to LangID is to build a generic language identifier that aims to correctly identify the language of text without any information about the source of the text. This approach has been fairly successful, and has resulted in a number of off-the-shelf language identifiers (Section 2.4) which are pre-packaged with general models for a number of languages, and enjoy widespread usage in research and commercial applications (Section 2.3). Some work has specifically targeted LangID across multiple domains, learning characteristics of languages that are consistent between different sources of text (Lui and Baldwin 2011). However, there

are often domain-specific features that are useful for identifying the language of a text. In this thesis, our primary focus is on LangID of digitally-encoded text, using only the text itself as evidence on which to base the prediction of language. Within text, there can sometimes be domain-specific peculiarities that can be used for LangID. For example, Mayer (2012) investigates LangID of user-to-user messages in the eBay e-commerce portal. He finds that using only the first two and last two words of a message is sufficient for identifying the language of a message. Another example of domain-specific textual evidence is the use of `xml:lang` attributes or the `<div lang="en">` construct, but relying on these alone suffers the potential pitfall that they are often incorrectly set (Rehurek and Kolkus 2009; Chew *et al.* 2011).

In many of the applications of LangID, the text being processed is accompanied by a variety of metadata, such as user identifiers, content headers, and timestamps. This information can be used in conjunction with text-based LangID to improve per-document LangID accuracy in domains that support it. Mayer (2012) reports that LangID accuracy over user-to-user messages can be further improved by including information about the site language, which can be obtained from the user profile. Bosca and Dini (2010) make use of the interface language of a user's web browser to help identify the language of search engine queries, finding that this can help boost LangID accuracy relative to just using the query text itself. LangID of web pages has also benefited from the use of domain-specific features. Martins and Silva (2005) make use of a number of heuristics specific to web pages, such as increasing the weight of text from certain regions of the document, and removal of boilerplate text

inserted by content management systems. They also make use of information from the URL of the document, and a similar filter is implemented in `ChromeCLD` (Sites 2013b). Baykan *et al.* (2008) carry out LangID of web pages using *only* the URL, ignoring the document content. They use the top-level domain, words in the URL, character $n$-grams of the URL as features and apply a naive Bayes classifier, as well as a meta-learning approach based on decision trees.

In addition to approaches that only consider document-level metadata, there are also approaches that consider information across a collection of documents. For example, Lui and Baldwin (2014) use user identity to identify Twitter users that only post messages in a single language, and use this information to build a collection of language-labeled Twitter messages. This form of collective classification can also be understood in terms of domain-specific priors. For example, Carter *et al.* (2013) use metadata from Twitter messages to compute language priors, which are then combined with a text-based model to produce a final language prediction. They make use of authorship, hyperlinks and dialog structure to produce per-language priors for any given message, and find that this method substantially improves on the LangID performance of text-only methods.

In this section, we have used the term "document" fairly loosely. According to the domain, a document could be a web page, a blog post, a Twitter message, or even all the content posted by a single user. While the exact metadata available varies by domain, there are some common trends, such as the availability of user identifiers and timestamps. Future work in this area could focus on comparing the effectiveness of different types of metadata in different application domains. Another

question of interest is the relative value of per-document metadata versus collection-level priors. The latter requires access to a large document sample to compute, and can be computationally expensive, begging the question: does it actually provide much utility beyond simpler models of the metadata available with each message, e.g. user and message metadata associated with individual Twitter messages? Once again, research on user geolocation would suggest yes, as metadata has been shown to be a stronger predictor of geolocation than the content of the message itself (Han *et al.* 2014b). Another open question is how textual and non-textual data can be integrated in a single language identifier, where work on meta-learning (Wolpert 1992; Dietterich 2002) and multi-task learning (Evgeniou and Pontil 2004; Jiang 2009) may be relevant.

### 2.5.9 Standardized corpora for LangID evaluation

As we discussed in Section 2.2.4, objective comparison of different methods for LangID is difficult due to the variation in the data that different authors have used to evaluate LangID methods, an open issue in LangID research (Hughes *et al.* 2006). Baldwin and Lui (2010a) emphasize the point by showing how the performance of a system can vary according to the data used for evaluation. This implies that comparisons of results reported by different authors may not be meaningful, as a strong result in one paper may not translate to a strong result on the dataset used in a different paper. In other areas of research, authors have proposed standardized corpora to allow objective comparison of different methods, for example the Reuters-21578 corpus (Lewis 1997) for topic-based text classification, or the OHSUMED corpus (Hersh

*et al.* 1994) for information retrieval in biomedical text.

In LangID research to date, the ECI multilingual corpus (Armstrong-Warwick *et al.* 1994) is perhaps the most commonly-used (Grefenstette 1995; Sibun and Reynar 1996; Elworthy 1998; Resnik 1999; Poutsma 2002; Da Silva and Lopes 2006; Takçı and Güngör 2012). Comparability of results is still questionable, because different authors have used different portions of the corpus. Furthermore, this corpus is proprietary, somewhat limiting widespread adoption. Another proprietary corpus that has been used in LangID research is the Reuters Corpus, Volume 2[12] (Vojtek and Bieliková 2007; Gottron and Lipka 2010; Lui and Baldwin 2011). An alternative source of language-labeled documents is parallel corpora used in MT, such as JRC-Acquis (Steinberger *et al.* 2006), which has been used by a number of authors (Konstantopoulos 2007; Lui and Baldwin 2011; Bergsma *et al.* 2012), or Europarl (Koehn 2005), which is also commonly used (McNamee 2005; Brown 2012; Lui and Baldwin 2012; Milne *et al.* 2012; Carter *et al.* 2013). Other data sources from which datasets for LangID have been derived are bible translations (Windisch and Csink 2005; Hammarström 2007; Chew *et al.* 2011; Brown 2012; Brown 2013), Wikipedia (Windisch and Csink 2005; Grothe *et al.* 2008; Rehurek and Kolkus 2009; Yang and Liang 2010; Baldwin and Lui 2010a; Chew *et al.* 2011; Lui and Baldwin 2011; Brown 2012; Bergsma *et al.* 2012; Majliš 2012; Milne *et al.* 2012; Brown 2013; Goldszmidt *et al.* 2013), translations of the Universal Declaration of Human Rights (Vatanen *et al.* 2010; Chew *et al.* 2011; Yamaguchi and Tanaka-Ishii 2012; King and Abney 2013), and the University of Oxford Text Archive (Souter *et al.* 1994). How-

[12] `http://trec.nist.gov/data/reuters/reuters.html`

| Reference | Source |
|---|---|
| Tromp and Pechenizkiy (2011) | Twitter |
| `http://www.win.tue.nl/~mpechen/projects/smm/LIGA_Benelearn11_dataset.zip` | |
| Brown (2013) | Bible Translations, Wikipedia |
| `http://sourceforge.net/projects/la-strings/files/Language-Data/` | |
| Tiedemann and Ljubešić (2012) | News Texts |
| `https://bitbucket.org/tiedemann/blacklist-classifier` | |
| Baldwin and Lui (2010a) | Government Documents, News Texts, Wikipedia |
| `http://www.csse.unimelb.edu.au/research/lt/resources/naacl2010-langid/` | |
| Baldwin and Lui (2010b) | Wikipedia (synthetic multilingual docs) |
| `http://people.eng.unimelb.edu.au/tbaldwin/etc/altw2010-langid.tgz` | |
| Zaidan and Callison-Burch (2011) | Online Commentary |
| `http://www.cs.jhu.edu/~ozaidan/AOC` | |
| Lui and Baldwin (2011) | Various |
| `http://people.eng.unimelb.edu.au/tbaldwin/etc/ijcnlp2011-langid.tgz` | |
| Majliš (2012) | Wikipedia |
| `http://ufal.mff.cuni.cz/~majlis/yali/` | |
| King and Abney (2013) | Web Crawl |
| `http://www-personal.umich.edu/~benking/resources/mixed-language-annotations-release-v1.0.tgz` | |
| Lui and Baldwin (2014) | Twitter |
| `http://people.eng.unimelb.edu.au/tbaldwin/data/lasm2014-twituser-v1.tgz` | |
| Lui *et al.* (2014) | Wikipedia (synthetic multilingual docs) |
| `http://people.eng.unimelb.edu.au/tbaldwin/etc/wikipedia-multi-v5.tgz` | |
| Tan *et al.* (2014) | News Texts |
| `https://bitbucket.org/alvations/dslsharedtask2014` | |

Table 2.3: Published LangID datasets

ever, these cannot be considered standardized corpora as authors have used different versions, subsets and splits of the data sources, and authors seldom release the exact dataset used.

Some authors have released datasets to accompany their work, to allow for direct replication of their experiments and encourage comparison and standardization. Dunning (1994) reports that the data used would be made available through the "Consortium for Lexical Resources", and that the dataset of Cavnar and Trenkle (1994) would be similarly available. However, as of writing, the "Consortium for Lexical Resources" appears defunct, and furthermore since 1994, no other work has been

published on either dataset. One dataset that has seen some re-use is the collection of Twitter messages in six languages published by Tromp and Pechenizkiy (2011), which has been used to evaluate LangID on Twitter messages (Lui and Baldwin 2012; Vogel and Tresner-Kirsch 2012; Goldszmidt *et al.* 2013; Lui and Baldwin 2014).

Table 2.3 lists a number of datasets that have been released to accompany specific LangID publications. In this list, we only include corpora that were prepared specifically for language identification research, and that include the full text of documents. Corpora of language-labelled Twitter messages that only provide document identifiers are also available (Bergsma *et al.* 2012; Carter *et al.* 2013; Zubiaga *et al.* 2014), but reproducing the full original corpus may be an issue as the original Twitter messages are deleted or are made otherwise unavailable (Lui and Baldwin 2014).

To address specific sub-problems in LangID, a number of shared tasks have been organized (see Table 2.4) on problems such as LangID in multilingual documents (Baldwin and Lui 2010b), code-switched data (Solorio *et al.* 2014), and discriminating between closely related languages (Zampieri *et al.* 2014). The datasets for shared tasks have generally been made publicly available after the conclusion of the task, and are a good source of standardized evaluation data. However, the shared tasks to date have generally targeted specific sub-problems in LangID, and no general, broad-coverage LangID datasets have been compiled.

One challenge in standardizing datasets for LangID is that the codes used to label languages are not fully standardized, and a large proportion of labeling systems only cover a minor portion of the languages used in the world today (Constable and Simons 2000). Xia *et al.* (2010b) discuss this problem in detail, listing different

| Title | Reference |
|---|---|
| Australasian Language Technology Workshop 2010<br>`http://comp.mq.edu.au/programming/index.htm` | Baldwin and Lui (2010b) |
| Twitter Language Identification Workshop at SEPLN 2014<br>`http://komunitatea.elhuyar.org/tweetlid/?lang=en_us` | Zubiaga *et al.* (2014) |
| First Workshop on Computational Approaches to Code Switching<br>`http://emnlp2014.org/workshops/CodeSwitch/call.html` | Solorio *et al.* (2014) |
| VarDial Workshop at COLING 2014<br>`http://corporavm.uni-koeln.de/vardial/sharedtask.html` | Zampieri *et al.* (2014) |

Table 2.4: Shared tasks and accompanying datasets

language code sets, as well as the internal structure exhibited by some of the code sets. Some standards consider certain groups of "languages" as varieties of a single macro-language, whereas others consider them discrete languages. An example of this is found in South Slavic languages, where some language code sets refer to Serbo-Croatian, whereas others make distinctions between Bosnian, Serbian and Croatian (Tiedemann and Ljubešić 2012). The unclear boundaries between such languages make it difficult to build a reference corpus of documents for each language, or to compare language-specific results across datasets.

Another challenge in standardizing datasets for LangID is the great deal of variation that can exist between data in the same language. We examine this in greater detail in Section 2.5.7, where we discuss how the same language can use a number of different orthographies, can be digitized using a number of different encodings, and may also exist in transliterated forms. The issue of variation within a language complicates the development of standardized datasets, due to challenges in determining which variants of a language should be included. Since we have seen that the performance of LangID systems can vary per-domain (Baldwin and Lui 2010a), that

LangID research is often motivated by target applications (see Section 2.3), and that domain-specific information can be used to improve accuracy (see Section 2.5.8), it often would not make sense to use a generic LangID dataset to develop a language identifier for a particular domain.

A third challenge in standardizing datasets for LangID is the cost of obtaining correctly-labeled data. Manual labeling of data is usually prohibitively expensive, as it would require access to native speakers of all the languages that the dataset aims to include. Large quantities of raw text data are available from sources such as the web or Wikipedia, but one problem in using this data is that it is frequently mislabeled (e.g. most non-English Wikipedias still include some English-language documents). In constructing corpora from such resources, it is common to use some form of automatic LangID (Scannell 2007; Stupar *et al.* 2011), but this makes such corpora unsuitable for evaluation purposes as they are already biased towards documents that can be correctly identified by automatic systems (Lui and Baldwin 2014). Lui and Baldwin (2014) propose the use of user identity to build a collection of Twitter messages for LangID evaluation that have not been directly selected through automatic means, but such a method is only applicable to domains where authorship information is available for documents. Future work in this area could investigate other means of ensuring correct gold-standard labels while minimizing the annotation costs.

Despite these challenges, standardized datasets would be very useful for promoting replicable and comparable research in language identification. Where a subset of data is used from a larger collection, researchers should include details of the specific subset, including any breakdown into training and test data or partitions for cross-validation.

Where data from a new source is used, justification should be given for its inclusion, as well as some means for other researchers to replicate experiments on the same dataset.

## 2.6 Chapter Summary

In this chapter, we presented a review of the relevant literature in LangID. The main theme of this review was to compare work-to-date in terms of several key aspects, including text representation (Section 2.2.1, Section 2.2.2), learning algorithms (Section 2.2.3) and evaluation (Section 2.2.4). We found that most of the relevant research in the area has independently revolved around a few key ideas. We also took a look at contexts where LangID has been applied (Section 2.3), and listed a number of attempts to build "off-the-shelf" LangID tools (Section 2.4). The focus of this thesis is on the construction and evaluation of such a generalized language identifier, and in the next chapter we will focus on one key aspect of this effort: the collection and preparation of language-labeled data from multiple sources.

# Chapter 3

# Data Collection for LangID Training and Evaluation

In Chapter 2, we observed that LangID research has often been application-driven, and as a result authors have tended to focus on data from specific sources, such as focusing on LangID of web pages or of Twitter messages. In more general work, authors have still tended to use data from a single source, and research has shown that existing approaches may not generalize across different data sources (Baldwin and Lui 2010a), a question we investigate in more detail in Chapter 4.

In order to develop a LangID system that is accurate regardless of characteristics or peculiarities of text from a particular source, we make use of data from a variety of sources. This will allow us to identify the characteristics of each language that are indicative of the language *independently* of the source that the data is drawn from. By maximizing the variation between the sources, we maximize our ability to identify the general characteristics of languages that we can exploit to achieve generalized

LangID. Maximizing variation in our data sources is also critical for evaluating a LangID system, as we need to show that the system is robust across the types of variation found in our data sources.

In the abstract, an ideal language identifier would be able to accept any type of document and output a label describing the language of the document content. This mimics the ability of a native speaker of each language, which is able to recognize his or her own language in various spoken and written forms. However, for the purposes of this thesis, we assume that the document is represented in text form in some machine readable and human interpretable encoding, though we do not assume that the actual character encoding (e.g. ASCII or UTF8) is known in advance. In other words, in this thesis we only deal with LangID for digital text. We explicitly exclude audio documents and images of text documents from consideration. Furthermore, we will initially focus on monolingual documents, i.e. documents that we assume to contain text from only one language. LangID for multilingual documents (documents containing text in more than one language) is an open issue in LangID research (Section 2.5.4), and we will examine the issue in greater detail in Chapter 6, but in this chapter we maintain the *monolingual assumption*, i.e. we will assume that each document contains text from only one language.

We begin this chapter with a discussion of the sources of variation between documents in the same language (Section 3.1). Thereafter, we identify a number of sources of data that we utilize to build datasets for the purposes of this thesis. For each source, we describe its characteristics, and relate it to similar sources that have been used in work to date on LangID. From each source, we construct a dataset for use

in our experiments. We give details of how this is done, and statistics of the datasets prepared. The final data collection that we use in this thesis consists of about 200,000 documents across 9 sources, totaling over 2.3 GB of data in 145 languages, with each language appearing in at least two different sources.

## 3.1 Intra-lingual Variation between Documents

Text in the same language can vary between different sources in a number of ways. One such source of variation that we previously identified in Section 2.5.7 is the document encoding. A document's encoding is a convention for mapping from the symbols used by a written language to a series of byte sequences used by a computer to represent the symbols for storage and processing. For any language, it is possible that it will exist in different encodings. The reasons for this may be practical (e.g. Unicode encodings such as UTF8 are seldom used on the Chinese web due to their relative inefficiency at representing Chinese text when compared to Guobiao standards and Big5), political (e.g. Guobiao is the official standard of mainland China whereas Big5 is the official standard of Taiwan), or historical (e.g. the use of EBCDIC rather than ASCII on IBM mainframes).

Encoding detection can be handled independently of LangID, where the encoding of a document is detected and the document is transcoded before attempting LangID (Kikui 1996), or it can be carried out concurrently, by detecting language-encoding pairs rather than just the language (Brown 2013). Research to date has tended towards joint detection of language and encoding, because encoding detection without knowing the underlying language is difficult, and furthermore an explicit encoding

detection and transcoding stage is relatively costly from a computational perspective. However, normalizing the encoding affords some advantages such as simplified models of each language, as well as the ability to leverage linguistic and other metadata provided by resources such as the Unicode Character Database (see Section 2.5.1).

For the purposes of this thesis, we chose not to normalize the encoding of the data we collect, in order to avoid the potential introduction of noise as a result of faulty encoding detection. Furthermore, including the documents in unnormalized form allows for more detailed investigation of the impact of encoding normalization, which would not be possible if the variation due to encoding were eliminated in advance. In practice, the actual impact of variance due to encoding is minimal as most of our data sources use a single encoding (see Table 3.1 on page 91). For 7 of our 9 sources, all documents collected are encoded in UTF8. For the remaining two sources, we either collected encoding information from included metadata (as in the case of Debian), or detected it using off-the-shelf tools where no such metadata is available (as in the case of CommonCrawl).

In addition to encoding, another non-linguistic source of variance between documents from different sources is in the markup used to represent information such as the formatting of the document. For web data, this is typically HTML, which is used to describe document structure and embed non-linguistic content such as images and tables. A closely related form of markup is XML, which is generally used to represent semi-structured data, which may then include free-text components. XML formats are a popular option for the storage and interchange of corpora used for linguistic research, as the availability of standardized parsers and querying mechanisms is per-

ceived to increase the accessibility of such data.[1] Despite being conceptualized as markup formats that are easy for both humans and machines to interpret, XML-like formats (such as HTML) are often found to be cumbersome due to their relatively high proportion of markup relative to the content they encapsulate. This has led to the development of "lightweight" markup formats, which are intended to be formal enough to be machine-readable while maintaining aesthetic properties that make them more suitable for human editors. One such format is WikiMarkup, used by the MediaWiki software that powers websites such as Wikipedia.

Markup format introduces several challenges for LangID that have not been previously addressed in the literature. Firstly, the use of markup introduces repetitive character sequences into text, which can cause complications for LangID systems that use term frequency (e.g. Cavnar and Trenkle (1994)). Secondly, the markup may introduce particular features that are strongly predictive of a language, which however are highly specific to the form of markup. For example, HTML allows a `lang` attribute to be associated with the HTML tag encapsulating a document. However, Chew *et al.* (2011) report that in a sample of 1660 web pages, only 698 (42.0%) were found to contain the attribute, and furthermore in 27.5% of these, the `lang` attribute was incorrectly set. In total, only 506 of 1660 web pages (30.5%) had a correct `lang` attribute. Finally, whereas the document markup itself is not part of the linguistic content of the document, it is often the case that the markup uses English words (e.g. the use of `head` or `body` in HTML). This can lead to false positives in LangID, where the language detected is that used by the markup rather than the document content.

[1] Whether this is true is well beyond the scope of this thesis

For reasons similar to those given for encoding, we have generally chosen not to normalize our source documents for markup. The case for not normalizing on the basis of markup is stronger than that for encoding, because whereas a document is meaningless unless interpreted under some encoding, markup is independent of the textual content of the document. While markup is generally meant to be machine-readable and hence relatively easy to eliminate, this pre-supposes accurate detection of the markup, which in turns requires a-priori knowledge of all the forms of markup to be eliminated. Furthermore, content extraction is in itself an unsolved problem and an active area of research for mining data on the web (Gupta *et al.* 2003; Song *et al.* 2004; Weninger *et al.* 2010; Sun *et al.* 2011). Given these issues, it is desirable to quantify how well existing systems respond to the noise introduced by markup, as well as to develop methods that are robust to markup. Thus, we do not normalize our source documents for markup, and furthermore specifically target variety in the types of markup we consider.

Related to the issue of encoding is variation in script. This can result from cultural reasons (e.g. the use of Hiragana, Katakana and Kanji in different contexts in Japanese), geo-political reasons (e.g. Traditional and Simplified Chinese), and also from transliteration (e.g. the use of both Cyrillic and Latin scripts in Serbian). From the perspective of modeling languages as discrete classes and then identifying the language of a document based on the class it most resembles, this can pose problems as without normalization for script, a clustering of documents may reveal that a language has multiple distinct centroids for different orthographies, and thus systems that model a language as a single "point" (in a vector-space model) or a single

distribution (in a probabilistic model) may end up with a representation that falls "between" the multiple modes and is a poor fit for any of them.

So far, we have only discussed non-linguistic reasons for variation in language between different sources of documents. There are also a number of linguistic reasons why language may vary. One obvious reason is that documents from different sources are likely to discuss different topics, and so the set of content words used is likely to be different. For example, bible texts will often make reference to religious terms and concepts, but the frequency of such lexical items is likely to be reduced in technical documentation for software. Hence, whereas the language-specific word for *God* may be a good predictor of language in bible translations, it may not be as effective in determining what language the documentation for a web browser is written in.

Register is another source of linguistic variation. The type of language used in government documents is dissimilar to that used in online forums, and this in turn can have an impact on what features are predictive of a language in each medium. Research comparing language models derived from different sources, from social media through to representative corpora, has found that there is a continuum in similarity between a variety of sources, with Twitter messages and a balanced corpus of British English on opposite ends (Baldwin *et al.* 2013). Partly related to register is regularity in document structure. Certain document sources have a very regular structure. For example, the Universal Declaration of Human Rights (UDHR) is a legal document, and consists of 30 discrete *articles* (article in the legal sense of the word, as a separate clause of paragraph of a legal document). In Section 2.5.9, we identified translations of the UDHR as a popular source of training and test data for LangID. Each article of the

UDHR is explicitly labeled, making the translation of the term *article* fairly predictive of each language. For example, Malay and Indonesian are known to be closely-related languages (Ranaivo-Malancon 2006), but *article* is translated to *perkara* in Malay and *pasal* in Indonesian. Bosnian, Serbian and Croatian are also known to be closely-related (Tiedemann and Ljubešić 2012), but *article* is translated to *član* in Bosnian and Serbian an *ćlanak* in Croatian. The term *article* appears frequently in the UDHR, and so translations of it are strongly predictive of language for samples of the UDHR, but the term is much less frequent and thus much less predictive in text from other sources.

User-generated content presents particular challenges (Eisenstein 2013), due to the informal register generally leading to much more extensive orthographic variation, either because of accidental misspellings or deliberate variations of lexical forms (Han *et al.* 2013). A commonly-observed phenomenon in languages that support casing is the use of case variation to convey emphasis – text can be written in ALL UPPERCASE to convey importance or emotion. Another phenomenon that is frequently observed in user-generated content is the loss of diacritics in languages that use them (e.g. Spanish or Czech).

## 3.2 Data Sources

As we discussed at the start of this chapter, to develop a generalized LangID system we require datasets that maximize variation in individual languages, so that we may focus on determining characteristics of each language that are independent of the source-specific variation. In the rest of this chapter, we describe a number of data

| Name | Type | Format | # Docs | # Langs | # Encodings | Size (MB) |
|---|---|---|---|---|---|---|
| JRC-Acquis | Legal Documents | Text | 20000 | 22 | 1 | 369.6 |
| Bible | Book | Text | 62892 | 65 | 1 | 284.6 |
| CommonCrawl | Webpage | HTML | 24096 | 42 | 22* | 1054.0 |
| Debian | Technical | Text | 21735 | 89 | 21 | 268.0 |
| RCV2 | Newswire | XML | 20000 | 12 | 1 | 67.7 |
| SETimes | Newswire | Text | 31551 | 10 | 1 | 147.6 |
| UDHR | Legal Documents | Text | 1270 | 127 | 1 | 2.1 |
| Wikipedia | Encyclopedia | WikiMarkup | 28600 | 143 | 1 | 99.4 |
| Twitter | Microblog | Text | 14178 | 65 | 1 | 15.0 |

* detected with chardet

Table 3.1: Statistics of datasets prepared for this thesis.

sources and the sources of variation that they capture, and relate them to work to date on LangID. We also describe the dataset that we construct from each source for the purpose of this thesis. A summary of the datasets prepared is given in Table 3.1.

### 3.2.1 Debian Internationalization

The Debian Project[2] is a worldwide organization of volunteers that maintains a well-known free and open source operating system. The members of Debian write and speak a wide variety of languages, and Debian maintains technologies and resources to make software available to users worldwide in their native language, a process commonly known as *internationalization* (abbreviated as i18n). A great deal of effort goes into the implementation of i18n, and one of the resources generated is databases of translations of interface messages. Debian uses the `gettext` system to manage these translations. In the `gettext` system, strings containing messages to be displayed to the user are wrapped in a function call to a special function provided by `gettext`. This allows `gettext` to build a list of strings that require translation, which are

[2] `http://www.debian.org`

| | |
|---|---|
| English | Display some available commands at the top of the screen |
| French | Afficher en haut de l'écran certaines des commandes disponibles |
| Italian | Mostrare alcuni comandi disponibili in cima allo schermo |
| Chinese | 将某些可用的命令显示在屏幕顶端 |

Figure 3.1: Example translations from the DEBIAN i18n database for the `aptitude` software package.

then presented separately to a human translator. The human translator provides a string-by-string translation, which is then stored in a translation database that can be distributed separately from the software itself. When the software runs, `gettext` selects the appropriate translation database based on the user's local configuration, and uses that to present the user with interface messages in the user's preferred language.

The translation databases are essentially sentence-aligned parallel texts. Figure 3.1 gives an example of a string translated from English into 3 other languages. Other researchers have used similar translation databases from open-source software as parallel texts for machine translation purposes (Tiedemann 2012). In our work, we do not make use of the parallel nature of the texts, and instead treat all the translated strings for a particular software package into a particular language as a single document. Common practice in `gettext` is to use English as the pivot language (i.e. the software developer will write all the embedded messages in English, and `gettext` is used to manage messages in all other languages). As a result of this practice, translation databases are packaged on a per-language per-software basis, and such a package contains parallel strings in English and the target language.

The Debian Project makes all of these translation databases available online free-

of-charge. We downloaded a copy of all the available databases on 17/02/2011. We only processed databases where the language code corresponded to a valid ISO639-1 code, resulting in 21,735 documents in 90 languages across 15 reported encodings.

### 3.2.2 JRC-Acquis

JRC-Acquis (Steinberger *et al.* 2006) is an aligned multilingual parallel corpus covering the 20 official languages of the European union, as well as additional languages from candidate countries. It totals over 460,000 documents, with an average of 8000 documents per language, corresponding to an average of about 9 million words per language. It is derived from the *EU/EC Acquis Communautaire*, which is the body of common rights and obligations with which all members of the European Union (EU) are bound. Thus, it primarily consists of legal documents translated into all the official languages of the EU. Each document in JRC-Acquis comes with alignment information for all languages it is available in, though not all documents are available in all languages. In this work, we do not make use of the alignment information.

To build our dataset for LangID, we randomly sampled 20,000 documents from the full collection, maintaining the relative skew of document counts between languages. For each document, only the contents of the XML tag `<body>` were retained. This eliminates some of the variation due to the use of XML markup, but was necessary because the documents in languages other than English may contain English metadata. Instead, we opted to build a dataset from this source that would place greater emphasis on the variation due to the style of language (legal documents).

```
Commission Regulation (EC) No 406/2005 of 10 March 2005
fixing the maximum export refund on common wheat in connection
with the invitation to tender issued in Regulation (EC)
No 115/2005

Commission Regulation (EC) No 406/2005
of 10 March 2005
fixing the maximum export refund on common wheat in connection
with the invitation to tender issued in Regulation (EC) No 115/2005
THE COMMISSION OF THE EUROPEAN COMMUNITIES,
Having regard to the Treaty establishing the European Community,
Having regard to Council Regulation (EC) No 1784/2003 of 29 September
2003 on the common organisation of the market in cereals [1], and in
particular Article 13(3) thereof,
Whereas:
(1) An invitation to tender for the refund for the export of common
    wheat to certain third countries was opened pursuant to Commission
    Regulation (EC) No 115/2005 [2].
(2) In accordance with Article 7 of Commission Regulation (EC) No
    1501/95 of 29 June 1995 laying down certain detailed rules for the
    application of Council Regulation (EEC) No 1766/92 on the granting
    of export refunds on cereals and the measures to be taken in the
    event of disturbance on the market for cereals [3], the Commission
    may, on the basis of the tenders notified, decide to fix a maximum
    export refund taking account of the criteria referred to in
    Article 1 of Regulation (EC) No 1501/95. In that case a contract
    is awarded to any tenderer whose bid is equal to or lower than the
    maximum refund.
(3) The application of the abovementioned criteria to the current
    market situation for the cereal in question results in the maximum
    export refund being fixed.
(4) The measures provided for in this Regulation are in accordance with
    the opinion of the Management Committee for Cereals,
HAS ADOPTED THIS REGULATION:
Article 1
For tenders notified on 4 to 10 March 2005, pursuant to the invitation
to tender issued in Regulation (EC) No 115/2005, the maximum refund on
exportation of common wheat shall be 10,00 EUR/t.
Article 2
This Regulation shall enter into force on 11 March 2005.
This Regulation shall be binding in its entirety and directly applicable
in all Member States.
```

Figure 3.2: Example document from JRC-Acquis dataset.

Figure 3.2 shows an example English-language document from the JRC-Acquis dataset. JRC-Acquis documents tend to be long, and contain a large amount of source-specific 'noise', such as reference codes for other documents. There is also an unusually high volume of numbers and dates, which are not source-specific per-se but are not as common in other datasets. It also contains a small amount of leftover noise from the use of XML, such as the use of ` ` for non-breaking spaces. Overall, the number of languages covered is relatively small, with approximately 17MB of data available for each language, far exceeding typical estimates of how much data per-language is required to train a language identifier before there is no further gain in accuracy (Brown 2013).

### 3.2.3 Reuters Corpus V2

Reuters RCV2[3] consists of over 487,000 documents in 12 languages. Each document contains an individual news story written by a local reporter in their own language, and as such, unlike JRC-Acquis and Debian, the documents in the collection are not parallel. The use of newswire data is fairly common in LangID research. For example, the frequently-used ECI corpus (Armstrong-Warwick *et al.* 1994) contains a fair amount of newspaper text.

We build a LangID dataset by randomly sampling 20,000 documents from the full collection, maintaining the relative skew of document counts between languages. Notable features of this dataset are that it does not include any English documents, and that it includes documents from non-European languages (Chinese and Japanese).

[3] `http://trec.nist.gov/data/reuters/reuters.html`

The description of the RCV2 corpus lists "Latin American Spanish" as a separate language from "Spanish". For the purposes of building a LangID dataset, we ignore the "Latin American Spanish" component of Reuters RCV2, and exclusively use documents from the "Spanish" component of the corpus. In addition to the text of the story itself, documents also include some metadata. The text and metadata together are stored in an XML format, which we retain entirely unnormalized. This is in contrast to JRC-Acquis (Section 3.2.2), where metadata and XML markup was removed due to the high incidence of English-language metadata in non-English documents.

Figure 3.3 gives an example of a document from the RCV2 dataset. We retained the full XML markup, which is a source of a large amount of noise, due to the repetitive patterns which may hamper term-frequency based methods, but also due to the presence of English words as part of the syntax of XML. Furthermore, RCV2 documents contain a fairly large amount of non-linguistic metadata, such as identifier codes assigned to individual documents by Reuters editors. Another salient feature is metadata that is strongly predictive of language – the `dc.source` tag indicates the language of the data, but obviously this feature would not help in determining the language of data from a different source. These features combined make generalized LangID of RCV2 documents particularly challenging.

```
<?xml version="1.0" encoding="UTF-8"?>

<newsitem date="1997-04-23" id="root" itemid="281103" xml:lang="it">
<title></title>
<headline> Francoforte, lira fix a 996,01 su dm da 996,61. </headline>
<byline></byline>
<dateline></dateline>
<text>
<p> FRANCOFORTE, 23 aprile (Reuter) - La lira e&apos; stata fissata a
996,01 su marco al fixing della seduta di Francoforte, da 996,61 ieri.
</p>
<p> (c) Reuters Limited 1997. </p></text>
<copyright>(c) Reuters Limited 1997</copyright>
<metadata>
<codes class="bip:countries:1.0">
<code code="EURZ">
<editdetail action="confirmed" attribution="Reuters BIP Coding Group"
date="1997-04-23"></editdetail></code>
<code code="GFR">
<editdetail action="confirmed" attribution="Reuters BIP Coding Group"
date="1997-04-23"></editdetail></code>
<code code="WEURZ">
<editdetail action="confirmed" attribution="Reuters BIP Coding Group"
date="1997-04-23"></editdetail></code></codes>
<codes class="bip:topics:1.0">
<code code="M13">
<editdetail action="confirmed" attribution="Reuters BIP Coding Group"
date="1997-04-23"></editdetail></code><code code="M132">
<editdetail action="confirmed" attribution="Reuters BIP Coding Group"
date="1997-04-23"></editdetail></code><code code="MCAT">
<editdetail action="confirmed" attribution="Reuters BIP Coding Group"
date="1997-04-23"></editdetail></code></codes>
<dc element="dc.publisher" value="Reuters Holdings Plc"></dc>
<dc element="dc.datepublished" value="1997-04-23"></dc>
<dc element="dc.source" value="Reuters - Notizie in Italiano"></dc>
<dc element="dc.creator.location" value=""></dc>
<dc element="dc.creator.location.country.name" value=""></dc></metadata>
</newsitem>
```

Figure 3.3: Example document from RCV2 dataset.

### 3.2.4 CommonCrawl

Common Crawl is a project to build and maintain an open crawl of the web "that can be accessed and analyzed by everyone."[4] To date, the crawl covers over 6 billion discrete URLs. The data is made available to the public via Amazon's S3 service, from which it can be downloaded in Web Archive (WARC) format, divided into fixed-size chunks. Each record in a WARC archive contains the raw document, any headers from the HTTP session when it was downloaded, as well as additional metadata such as the URL from which it was downloaded, a timestamp and a unique identifier for the document with respect to the entire crawl. Common Crawl is meant to be a representative crawl of the Internet, and as such contains a highly diverse variety of content and languages. Most of the documents are in HTML format, though there are also a number of binary formats present such as PDF.

As we discussed in Section 2.3, LangID of web pages is an important application of LangID research. Having a dataset of web pages is thus important for investigating generalized LangID, from both training and evaluation perspectives. One problem that we faced is that there is a lack of a suitable corpus of language-labeled web pages. This is largely due to the fact that web crawls tend to happen on a massive scale, gathering millions or even billions of documents, thus making manual LangID impossible. The ClueWeb09 corpus (ClueWeb09 2009) is an example of a web crawl that has attempted to provide web pages pre-grouped by language. It consists of about 1 billion web pages in 10 languages. In ClueWeb09, the language of each document was automatically detected using `textcat`, which is described in detail in

[4] `http://www.commoncrawl.org`

Section 4.1.1. However, previous work has shown that the language labels of a fair number of the documents in ClueWeb09 are incorrect (Cook and Lui 2012). We thus opted to assemble a new dataset of web documents for LangID research.

To build a dataset for use in our LangID research, we downloaded 100 random chunks from Common Crawl, and extracted all the raw documents, discarding WARC metadata and HTTP session information. We then applied two off-the-shelf LangID systems: `langid.py`, a locally-developed system based on an early version of the research in this thesis (Lui and Baldwin 2012); and `ChromeCLD` (Sites 2013b), a standalone package of the language identifier embedded in the Google Chrome browser, which is optimized for LangID of web pages.

We base our construction of this dataset on a principle of high precision. We thus discarded any documents where the two systems did not agree on the language of the document (about 21% of documents). Disagreement could be due to one of the identifiers being wrong, but could also be due to a document being in a language outside the training set of one of the identifiers, or one of the identifiers reporting insufficient confidence to make a language prediction. After this initial filtering, we then discarded documents for any language with less than 50 documents. From the remaining documents, we sampled up to 1000 documents per language. For languages where less that 1000 documents were available, we used all the available documents. This left us with a final dataset consisting of 24096 documents in 42 languages.

Our CommonCrawl dataset is different from our other datasets in that the per-document language labels are automatically assigned. This process means that our CommonCrawl dataset is no longer a fully representative sample of a general web

crawl, as we have had to discard languages with insufficient data, and documents where the off-the-shelf language identifiers we used disagreed. We have done this so that we can expect the labels for the documents that we have included in our dataset to be largely correct, even if the labels are technically a "silver-standard" rather than a manually-labeled "gold-standard". The main compromise is that the dataset only contains documents that are relatively easy for language identifiers to correctly identify. This means that in absolute terms, the accuracy on this dataset is likely to be an overly optimistic estimate of the difficulty of LangID of web documents. However, in this thesis, we are not interested in the accuracy of LangID in any single specific domain as much as we are in comparing and improving LangID across multiple domains. For such purposes, this dataset is adequate, and we shall see in later chapters that we are able to use this data to illustrate key issues in training of generalized language identifiers.

Figure 3.4 shows an example document from the CommonCrawl dataset. The typical CommonCrawl document contains a substantial proportion of HTML markup, which carries with it similar technical issues to XML markup, namely repetitive sequences that can be problematic for term-frequency based approaches, as well as the use of a tagset that contains English words.

```
HTTP/1.1 200 OK
Date:Tue, 07 Sep 2010 03:22:23 GMT
Server:Apache/2.0.52 (Red Hat)
Last-Modified:Wed, 24 Oct 2007 23:24:14 GMT
ETag:"a1400a-29f-6a66c380"
Accept-Ranges:bytes
Content-Length:671
Connection:close
Content-Type:text/html
x_commoncrawl_OriginalURL:http://scea2000.org/Webpage1/necrology/index.html
x_commoncrawl_URLFP:5520245254828710242
x_commoncrawl_HostFP:1225043235280578374
x_commoncrawl_FetchTimestamp:1283829744469

<html>
<head>
<title>Necrology</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF" text="#000000" link="#FFFFFF" vlink="#FFFFFF">
<p align="center"><font color="#FF0000" size="7">Necrology</font></p>
<p>    The SCEA wishes to extend its heartfelt
condolences to any member who has lost a loved one.  If you have
lost a parent, child, or spouse the SCEA will send $20 to your school
library to purchase a book in memory of your loved one.  Please
notify the SCEA office (323-2131) if you or an SCEA member you know has
had a death in their family.</p>
</body>
</html>
```

Figure 3.4: Example document from CommonCrawl dataset.

### 3.2.5 Wikipedia

Wikipedia is an online encyclopedia maintained by a worldwide community of volunteer writers and editors. It is hosted by the Wikimedia foundation,[5] a "nonprofit charitable organization dedicated to encouraging the growth, development and distribution of free multilingual, educational content, and to providing the full content of these wiki-based projects to the public free of charge." Individual languages have their own independent Wikipedias, usually under the corresponding ISO 639-1 code. For example, English Wikipedia that most English-speaking Internet users are familiar with is accessible at `http://en.wikipedia.org`. Wikipedia provides static dumps of the complete contents of all Wikipedia wikis,[6] exported automatically following a rotating export schedule. The contents of these dumps are licensed under the GNU Free Documentation License and the Creative Commons Attribution-Share-Alike 3.0 License. Wikipedia provides massive quantities of textual data, with quality comparable to traditionally-edited encyclopedias (Giles 2005). This makes it a highly attractive source of data for research into many aspects of natural language, and one of the most popular sources of data for LangID research to date (Rehurek and Kolkus 2009; Baldwin and Lui 2010a; Winkelmolen and Mascardi 2011; Yamaguchi and Tanaka-Ishii 2012; Goldszmidt *et al.* 2013).

For the purpose of constructing a LangID dataset for this thesis, in December 2012 we obtained XML dumps of all Wikipedias with valid ISO 639-1 codes, giving us Wikipedia database exports for 180 languages. We discarded exports that contained less than 100 documents, and after construction of the remaining datasets described

[5] `http://wikimediafoundation.org/`

[6] `http://dumps.wikimedia.org/backup-index.html`

```
'''Flowers''' is an [[Unincorporated area|unincorporated community]] in
[[Warren County, Mississippi|Warren County]], [[Mississippi]]. It is
located approximately three miles east of [[Bovina, Mississippi|Bovina]]
and is part of the [[Vicksburg, Mississippi|Vicksburg]] [[Vicksburg
micropolitan area|Micropolitan Statistical Area]]. The Ceres Industrial
Park, one of many industrial areas in Warren County, is located in Flowers.

{{Warren County, Mississippi}}
{{Mississippi-geo-stub}}

{{coord missing|Mississippi}}

[[Category:Unincorporated communities in Mississippi]]
[[Category:Populated places in Warren County, Mississippi]]

[[vo:Flowers]]
```

Figure 3.5: Example document from the WIKIPEDIA dataset.

in this chapter we discarded any languages that did not appear in any other dataset. This left us with a total set of 143 languages. For each language, we randomly selected 1000 raw pages of at least 250 bytes in length (including markup, as we did not preprocess the data). For languages where less than 1000 such pages were available, we selected all the pages available for that language. The final dataset consists of 286000 documents.

Figure 3.5 shows an example document from the WIKIPEDIA dataset. One characteristic of the WIKIPEDIA data is the use of "wikimarkup", a language for encoding document structure that is machine readable yet relatively lightweight, allowing for easy editing by a human editor using a simple text editing program. In our example, we see examples of hyperlinking that wikimarkup uses to encode intra-Wikipedia links, as well as metadata, such as the "category" membership of a particular article.

### 3.2.6 Universal Declaration of Human Rights

The Universal Declaration of Human Rights (UDHR) is a document that lists rights that all human beings are inherently entitled to. It consists of a preamble that states the premise and general principles of the declaration, followed by 30 clauses (known in legalese as *articles*) that list the rights shared by all individuals by virtue of being human. It was first adopted by the United Nations General Assembly on 10 December 1948, and despite not being legally binding, has been highly influential in shaping policy and law worldwide. The Office of the High Commissioner for Human Rights maintains translations of the Universal Declaration of Human rights in 379 languages at time of writing,[7] and has been named by Guinness World Records as the most translated document in the world.[8] The text of each translation is available through various sources, including *UDHR in Unicode*,[9] a project to demonstrate the use of Unicode for a wide variety of languages.

Due to the relatively standardized quantity of data, as well as the diversity of languages covered, use of UDHR translations has been common in LangID research to date (Vatanen *et al.* 2010; Chew *et al.* 2011; Yamaguchi and Tanaka-Ishii 2012; King and Abney 2013). In this thesis, we use the *UDHR in Unicode* version of the UDHR translations as it avoids issues of encoding. Other versions of the UDHR translations present the document in image form for certain languages, which is unsuitable for our purposes. We only make use of the languages with ISO639-1 codes, a total of

[7] http://www.ohchr.org/EN/UDHR/Pages/Introduction.aspx

[8] http://www.guinnessworldrecords.com/world-records/1000/most-translated-document

[9] http://www.unicode.org/udhr/

```
Article 1
All human beings are born free and equal in dignity and rights. They are
endowed with reason and conscience and should act towards one another in a
spirit of brotherhood.

Article 2
Everyone is entitled to all the rights and freedoms set forth in this
Declaration, without distinction of any kind, such as race, colour, sex,
language, religion, political or other opinion, national or social origin,
property, birth or other status. Furthermore, no distinction shall be made
on the basis of the political, jurisdictional or international status of
the country or territory to which a person belongs, whether it be
independent, trust, non-self-governing or under any other limitation of
sovereignty.

Article 3
Everyone has the right to life, liberty and the security of person.
```

Figure 3.6: Excerpt from the English version of the Universal Declaration of Human Rights (UDHR).

138 languages. For each language, we divide the whole declaration into 10 segments with equal numbers of lines. Thus, in this dataset, there are 10 documents for each language, for a total of 1380 documents.

Figure 3.6 presents an excerpt from the English-language version of the UDHR, covering the first three articles of the declaration. The register of the text is formal, and the text itself is free from any intervening markup, except for the article headings. However, the article headings are highly regular, and provide a strongly characteristic feature for each language. This means that when dividing the document into sub-documents for use in training/evaluation setups such as cross-validation, results are likely to be artificially high due to this particular regularity. We highlighted some examples of this in Section 3.1, such as the translation of the English term *article*

(in the legal sense) to *perkara* in Malay and *pasal* in Indonesian, making it trivial to distinguish between subsections of the two translations.

### 3.2.7 Bible

The Bible is a collection of religious texts that is considered sacred in a number of interrelated faiths. In historical terms, the document has a complex history, and different denominations make use of different subsets of the texts as their own canonical version. Even where there is agreement between groups on which texts are sacred, there can be subtle variations in the particular translations used. Nonetheless, Bible-derived corpora are attractive for LangID research because they typically provide a reasonable amount of well-curated text. Translations are often prepared and maintained by religious organizations around the world as part of missionary efforts.

A number of previous authors have made use of text from Bible translations in LangID research (Hammarström 2007; Vatanen *et al.* 2010; Chew *et al.* 2011; Brown 2013). The source of Bible translations has varied, as has the set of languages covered. Gaining access to sets of Bible translations can be difficult, as the documents are sometimes published in unsuitable formats (e.g. PDF), or terms and conditions are applied to the access and use of the translations. In this thesis, we use a Bible Corpus assembled by Christos Christodoulopoulos.[10] We only use the bible versions for languages with ISO639-1 codes, for a total of 65 languages. Of these, 11 only have translations of the New Testament available, and a further 4 have only parts of the bible. We divide the text for each language into one document per chapter. We thus

[10] `http://homepages.inf.ed.ac.uk/s0787820/bible/`

```
In the beginning God created the heaven and the earth.
And the earth was without form, and void; and darkness was upon the face
of the deep.
And the Spirit of God moved upon the face of the waters.
And God said, Let there be light: and there was light.
And God saw the light, that it was good: and God divided the light from
the darkness.
And God called the light Day, and the darkness he called Night.
And the evening and the morning were the first day.
And God said, Let there be a firmament in the midst of the waters, and
let it divide the waters from the waters.
And God made the firmament, and divided the waters which were under the
firmament from the waters which were above the firmament: and it was so.
And God called the firmament Heaven. And the evening and the morning
were the second day.
And God said, Let the waters under the heaven be gathered together unto
one place, and let the dry land appear: and it was so.
And God called the dry land Earth; and the gathering together of the
waters called he Seas: and God saw that it was good.
```

Figure 3.7: Excerpt from an English document from the BIBLE dataset.

form a dataset containing 62892 documents.

Figure 3.7 shows an example from an English document in the BIBLE dataset. The text in this dataset is free of any intervening markup, and represents standardized language in a fairly formal register. The particular type of writing is also highly regular, as can be seen in Figure 3.7, which leads to what is likely to be a relatively 'easy' dataset for LangID. Furthermore, the topic of the writing leads to the prominence of certain nouns (the prominence of *God* is clearly visible in the example in Figure 3.7), which may be characteristic of each language within the same dataset, but may not generalize well across different datasets.

### 3.2.8 SETimes

SETimes (Tyers and Alperen 2010) is a parallel corpus of English and south-east European languages, based on content published on a Balkan news portal.[11] The original corpus covered 8 languages of the Balkan region and English, for a total of 9 languages. The corpus itself is structured similarly to the JRC-Acquis corpus (Steinberger *et al.* 2006), and is intended for use in machine translation and multilingual natural language processing research. An updated version of the corpus is included in OPUS (Tiedemann 2012), which corrects a number of existing issues in the original corpus, and adds data for Bosnian. A subset of the corpus has previously been used for research into language discrimination between Serbian, Bosnian and Croatian (Tiedemann and Ljubešić 2012).

In this work, we use the full corpus, which covers 10 languages (including English). The corpus is supplied in parallel format; we build our dataset by separating the parallel content back into the original documents. This yields 31551 documents across the 10 languages. Like the JRC-Acquis dataset which focuses on Western European languages, this corpus is limited in scope as it only covers Balkan languages. However, including such corpora allows us to study how to integrate different sources of data in building a language identifier, especially when the different sources each provide incomplete coverage of the overall language set.

The documents in the SETimes dataset consist of newswire articles, an example of which is shown in Figure 3.8. Being from a newswire source, the text is edited and curated, but exhibits greater diversity in style, structure and topics than text from

[11] `http://www.setimes.com`

```
The riders, called "Alkars", compete in traditional 18th century garb.
[Ksenja Jurkovic/SETimes] Every August in the small Croatian town of
Sinj, riders don 18th century garb to compete in a nearly three-century
old tournament popularly known as a Sinjska Alka. This unique
equestrian event is more than a tourist attraction. For many Croats, it
symbolises heroism, love for the homeland, and the struggle for freedom.
As the country prepares for EU membership, this and other examples of
Croatia's national heritage take on a new significance. Speaking at the
event, President Ivo Josipovic pledged that the country will "affirm its
identity and culture" and make a recognisable contribution to the
European tapestry. First Macedonian Eurobonds to be issued in December
Macedonia is set to issue its first Eurobonds by the end of the year, as
part of its overall strategy for raising its borrowing capacity and
attracting investors. Funds from the bond issue will go towards
restructuring the country's debt.
```

Figure 3.8: Example document from the SETimes dataset.

legal documents such as that present in JRC-Acquis.

### 3.2.9 Twitter

Twitter[12] is a microblogging platform with a worldwide user base, reporting 241 million active users sending over 500 million messages a day as of February 2014.[13] We discuss Twitter in more detail in Chapter 7, where we examine the performance of state-of-the-art LangID tools in this domain. Recent work has generated a number of Twitter-specific approaches to identifying the language of Twitter messages (Carter *et al.* 2013; Tromp and Pechenizkiy 2011; Bergsma *et al.* 2012; Goldszmidt *et al.* 2013), making it an interesting target for generalized LangID. In this section, we limit ourselves to giving a brief overview of the challenges of automatic LangID of Twitter messages, and a quick summary of the dataset we have constructed. In this thesis, we use a dataset of Twitter messages constructed using a "mostly-automated" approach to labeling Twitter messages by language, taking advantage of user identity to label the language of individual messages. A more detailed description of the procedure for constructing the dataset can be found in Chapter 7.

LangID of Twitter messages is challenging for a number of reasons. Messages are limited to 140 characters, which is a fairly small amount of data for statistical methods. The short message length and informal nature of Twitter messages has led to variations in the use of language that diverge from typical assumptions about vocabulary, spelling and syntax (Eisenstein 2013). This non-standard use of language on Twitter means that models of language derived from canonical corpora typically transfer rather poorly to Twitter messages (Baldwin *et al.* 2013). There is also a wide variety of languages present on Twitter – Bergsma *et al.* (2012) report observing 65

[12] `http://www.twitter.com`

[13] `https://about.twitter.com/company`

| | |
|---|---|
| English | RT @noafex: Make sure yall check out my homegirl @MsTorieSmith Blog!!! She's the lady Dr. Phill for you!!! lol http://t.co/d1LApys5 |
| French | UN GROOOOS AUREVOIR POUUUUR JULIEEE ! (@missjulie28 live on http://t.co/ZFJTmpkO) |
| Italian | La mia TL è stata bloccata dal maltempo o è da 10 minuti che nessuno se la sente di dire qualcosa? |
| Japanese | @mari37ml あら?ご近所さんだった♪( ´ ▽ ` ) 関西のファミリーがこっち来たら埼玉にしゅーごーだ! |
| Indonesian | Lagi dan lagi vote #film1 ah biar #PCWindows7 dri @saatpalingpas bsa ku dpat amin :) |

Figure 3.9: Examples of language-labeled Twitter messages.

languages in a 10M message sample, and Baldwin *et al.* (2013) report observing 97 languages in a 1M message sample. In both cases, the authors are only limited by the number of languages the tool used supported.

Figure 3.9 shows some examples of Twitter messages from our TWITTER dataset. In these messages, we can see some Twitter-specific features, such as "ReTweets" (RT), username references (@XXX), hash-tags (#XXX), as well as some features that are common in user-generated content, such as URLs and smilies. There is also an example of the use of capitalization for emphasis. All these sources of variation contribute towards making Twitter a harder target for generalized LangID.

## 3.3 Chapter Summary

In this chapter, we discussed issues regarding the preparation of datasets for experiments in generalized LangID. We began the chapter with an overview of sources of variation between documents in the same language, covering both linguistic and non-linguistic sources of variation. Thereafter, we identified 9 different sources of data

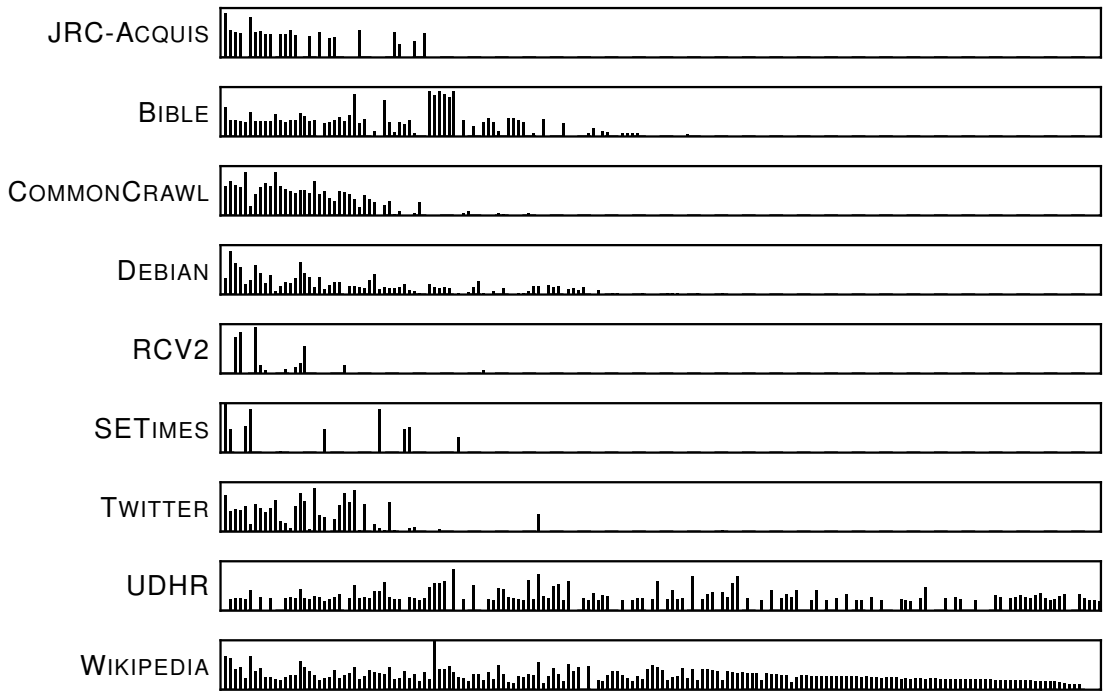



Figure 3.10: Relative quantity of data (in bytes) between languages for each dataset.

from which we prepared datasets of language-labeled documents. For each source, we gave a brief description of its characteristics, along with example content, with a particular emphasis on the types of variation present in the data from that source. We also described the method that we used to produce a dataset from each data source. The final document collection used in this thesis consists of about 200,000 documents across 9 sources, totaling over 2.3GB of data in 145 languages. Figure 3.10 provides visualization of the relative quantity of data for each language between different datasets. As noted in the chapter introduction, each language appears in at least two sources, but in some instances the quantity of data available is much smaller in one source than the other, and as such the smaller "spike" for these languages may not be visible in Figure 3.10.

In Chapter 4, we make use of the data we have collected to investigate the cross-domain generalizability of existing LangID systems. Thereafter, in Chapter 5 we develop a document representation for LangID that is robust to the types of variation we have seen in the data that we collected in this chapter.

# Chapter 4

# Cross-domain Generalizability of LangID Systems

Research to date has proposed a variety of methods for LangID, and we have identified and briefly discussed some of them in Chapter 2. Methods proposed have generally been evaluated independently of each other, and due to a lack of standardization in evaluation data and metrics, the published results are not directly comparable, which hinders objective evaluation. In previous chapters, we have discussed evaluation metrics proposed to date (Section 2.2.4), as well as suitable sources of data for training and evaluating LangID systems (Chapter 3). In this chapter, we undertake a systematic comparison of three existing LangID systems, two of

This chapter is based on work previously published as:

Lui, Marco, and Timothy Baldwin. 2011. Cross-domain Feature Selection for Language Identification. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, 553 – 561, Chiang Mai, Thailand.

which have previously been described in the literature (Cavnar and Trenkle 1994; Prager 1999a), and the third of which has been published online (Nakatani 2010a). In research to date, there have been a number of confounding factors that have limited the comparability of results. One factor that varies widely is the number of languages considered. In general, we would expect that considering more languages simultaneously should make the problem harder, as the relative proportion of data in any given language decreases. For example, Cavnar and Trenkle (1994) report almost perfect accuracy over the 8 languages they consider, but Baldwin and Lui (2010a) find that when the number of languages is increased, the languages with less training data tend to lag in accuracy (a general phenomenon in NLP problems involving classification that is sometimes referred to as "the long tail"). This is further complicated when we consider the "type" of languages investigated. Research to date suggests that samples of well-edited Western European languages are generally fairly easy to distinguish with letter trigram models (Cavnar and Trenkle 1994; Souter *et al.* 1994; Dunning 1994), but research has shown that this type of model does not suit closely-related Eastern European languages (Tiedemann and Ljubešić 2012). Another factor known to affect machine learning algorithms in general is the quantity of training data available, both in absolute terms as well as in relative quantities between classes. As this is not controlled for in previous work, the results are again made harder to compare. In this chapter, our aim is to provide a meaningful comparison of LangID accuracy between the three systems we have identified, and thus we train and evaluate each system on the same pairs of training and evaluation data, such that the difference in performance is only due to the system used, and not

confounding factors such as source and quantity of training data.

The data we use for training and evaluation is drawn from a variety of sources (Chapter 3), and is intended to capture variation within a language across the dimensions we discussed in Section 3.1, such as topic and register, as well as non-linguistic sources of variation such as encoding and domain-specific markup. For each of the text sources, we follow standard machine learning practice and divide all the available data into training (TRN), development (DEV) and test (TST) partitions. Documents are allocated to the partitions in a randomized fashion, stratified by language. The approximate ratio of sizes between the three partitions is 8:1:1 for training:development:test. The training partitions are used as training data for each system, and the development partitions are used for tuning any parameters that the system may have, such as an internal threshold or the number of features selected. Systems are then compared on the basis of their performance on the test partition, given the best parameters discovered using the development partition. This avoids any bias introduced by the system over-fitting to the development data. In this chapter, we are interested in investigating the generalizability of LangID models across the sources of variation that we have identified. In order to do this, we first establish a benchmark result *in-domain*, that is, drawing training and test data from the same text source. We use the terms "domain" and "text source" interchangeably, deferring a precise definition of "domain" to when we discuss transfer learning in more detail in Section 5.3.3.

In-domain evaluation corresponds to the usual formulation of LangID as a supervised machine learning problem (Cavnar and Trenkle 1994; Prager 1999a; Baldwin

and Lui 2010a). However, many LangID systems come with pre-trained models (see Section 2.4). Even when the system can be re-trained, language-labeled documents from the target text source may not be available, and so it is common to use easily-available sources of language-labeled text as a proxy. An example of this is the use of `TextCat`, an implementation of the method of Cavnar and Trenkle (1994) for LangID of web pages (ClueWeb09 2009).[1] Data from EU parliament datasets and newswire corpora were used to re-train `TextCat` before applying it to web pages crawled as part of ClueWeb09. In machine learning terms, this is an example of *transfer learning*, the use of data from external domains (in this case, a different text source) to improve task performance on a target domain. Pan and Yang (2010) provide a survey of transfer learning, in which they define *transductive* transfer learning, where labels are available in source domain(s) but not in the target domain. In-domain evaluation does not take into account the transfer learning effects, which we can expect to degrade accuracy, particularly if the source and target domains have some sort of systematic difference (Section 3.1).

To investigate the generalizability of LangID models across the sources of variation we have identified, we compare the in-domain results to the accuracy of the same systems in a *cross-domain* setting, that is, where the training and the test data are drawn from different sources. This allows us to quantify the penalty imposed by not having training data in a target domain, one of the aspects of generalized LangID we have identified in Chapter 1. We analyze the performance of each of three LangID systems in two cross-domain settings: (1) the training data is drawn from a single

[1] `http://boston.lti.cs.cmu.edu/clueweb09/wiki/tiki-index.php?page=Language+Identification+for+ClueWeb09`

| System | Described In | Document Representation | Classifier |
|---|---|---|---|
| `TextCat` | Cavnar and Trenkle (1994) | frequency profile | rank-order statistics |
| `Linguini` | Prager (1999a) | TF – IDF style | vector-space |
| `LangDetect` | Nakatani (2010b) | multinomial distribution | naive Bayes |

Table 4.1: Summary of LangID systems compared.

dataset and the test data is drawn from a single dataset that is different to the training dataset, and (2) the test data is drawn from a single dataset, but the training data is drawn from the union of all available datasets excluding that used for testing. These two settings serve to quantify the impact of the naive solution to "cross-domain" LangID, the first being to use labeled training data from a single dataset, and the second to pool training data across multiple datasets.

In this chapter, we make the simplifying assumption that all documents are monolingual, i.e. contain text in only one language. We return to the issue of LangID of documents that may contain text in more than one language in Chapter 6.

## 4.1 Systems Compared

The systems we are comparing are summarized in Table 4.1, and represent three different approaches to LangID. They were chosen on the basis that they are fairly well-known, their theoretical foundations are well-understood and each of them can be re-trained on new data. In this section, we give an overview of each system and its parameters, and a short evaluation of the similarities and differences between the systems.

### 4.1.1 TextCat

Cavnar and Trenkle (1994) describe a text classification algorithm based on rank-order statistics of letter sequences, and present an evaluation of it applied to LangID. The name `TextCat` does not appear in Cavnar and Trenkle (1994), but rather originates from the Perl implementation of Gertjan van Noord. Nonetheless, the name `TextCat` has come to be associated with the method of Cavnar and Trenkle (1994) as applied to LangID, and a number of other implementations have been given similar names. In this work, we use the name `TextCat` to refer to the method of Cavnar and Trenkle (1994) applied to LangID rather than the particular implementation of van Noord.

`TextCat` has been widely used in research settings where a LangID tool is required. Examples include language filtering for building minority language corpora (Ghani *et al.* 2004), web page LangID in the ClueWeb09 Dataset (ClueWeb09 2009), preprocessing of OpenSubtitles parallel data in OPUS (Tiedemann 2009:Section 2.1.1), and LangID of Twitter messages (Carter *et al.* 2013).

In `TextCat`, documents are first normalized by discarding digits and punctuation, keeping only letters, apostrophes and whitespace. The document is then tokenized into contiguous byte $n$-grams, using a mixture of $n$-gram orders ($1 \leq n \leq 5$). The frequency of each token is counted, and the representation of the document consists of all the $n$-grams in the document ranked by the frequency in which they occur in the document. Cavnar and Trenkle (1994) refer to this as the *n-gram profile* of the document (Figure 4.1). In a similar fashion, an $n$-gram profile is computed for each language from the respective training data. Figure 4.2 illustrates the process of com-

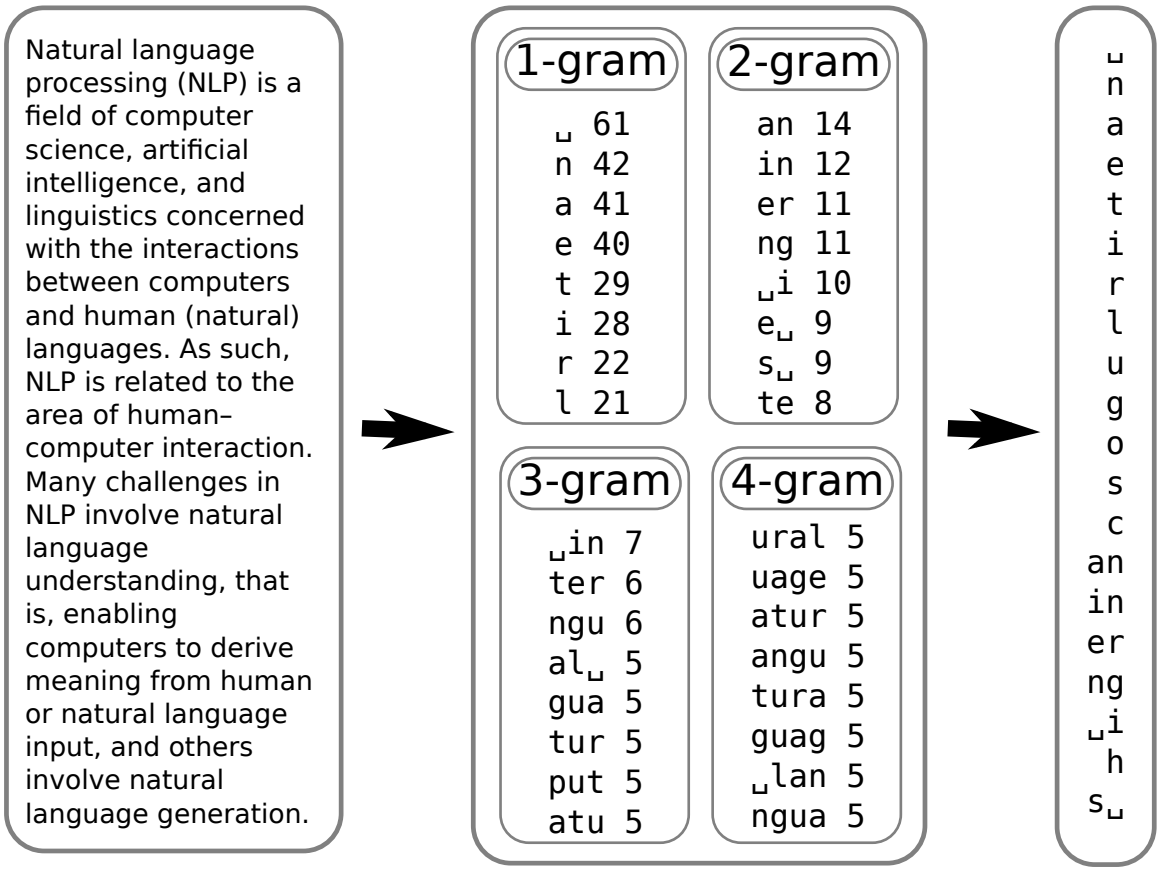



Figure 4.1: Document representation used by `TextCat`. Text is converted to a frequency distribution over letter $n$-gram sequences. Sequences are ranked by relative frequency and only the top-ranked sequences are retained.

puting language profiles for a toy dataset. The $n$-gram profile for each language is computed by summing the frequency of each $n$-gram across all the training documents for the language, or equivalently by concatenating all the documents for a language into a single "super-document", and then computing the $n$-gram profile thereof.

Cavnar and Trenkle (1994) report a number of characteristics of the language profiles they computed over a set of 8 Western European languages. In particular, they observe that:

| | **Class** | `an` | `in` | `er` | `ng` |
|---|---|---|---|---|---|
| doc1 | lang1 | 10 | 2 | 8 | 1 |
| doc2 | lang1 | 12 | 6 | 8 | 4 |
| doc3 | lang2 | 11 | 12 | 3 | 2 |
| doc4 | lang2 | 14 | 8 | 1 | 1 |
| — | lang1 | 22 | 8 | 16 | 5 |
| — | lang2 | 25 | 20 | 4 | 3 |

Figure 4.2: Deriving language profiles for `TextCat`. The vector for each language is the sum of the vectors for all the documents in the language.

- The top 300 $n$-grams are highly correlated to the language.
- The highest $n$-grams are mostly 1-grams, followed by function words, and then frequent prefixes and suffixes.
- Around rank 300 or so, the $n$-grams become more specific to the subject of the document.
- There is nothing special about rank 300; the value was selected by inspection.

`TextCat` decides on the language of an unlabeled document by comparing its $n$-gram profile to the profile of each language, and then labeling the document by selecting the best-scoring profile. Figure 4.3 illustrates the method for scoring a profile, dubbed the *out-of-place* metric by Cavnar and Trenkle (1994). This metric is a form of rank-order correlation, though the actual method described is ad-hoc rather than statistically motivated. In summary, the method measures the sum of the absolute differences in rank across the feature set. Lower scores are considered better, the intuition being that the most frequent features ordered by frequency should be similar in documents of the same language but different between different languages.

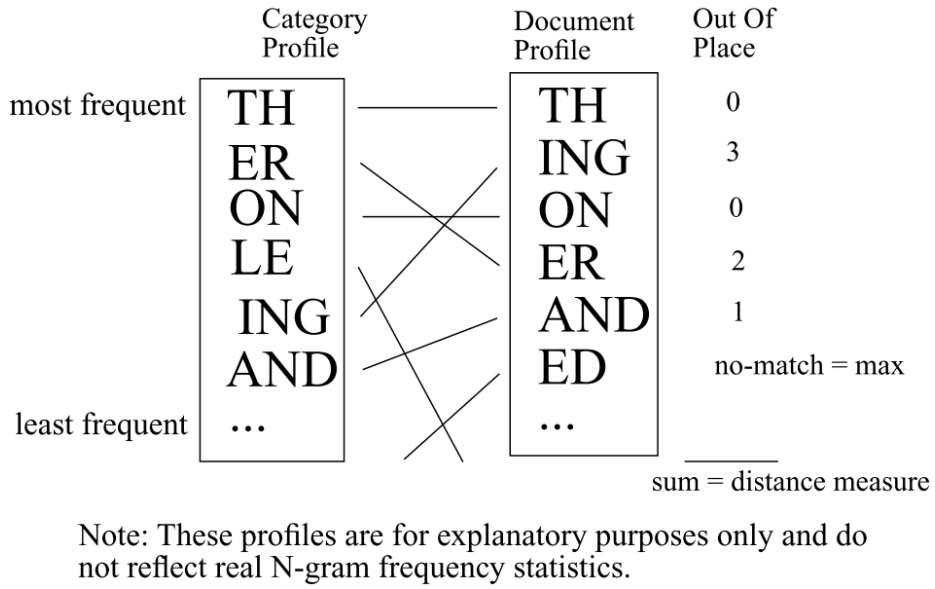



Figure 4.3: Calculation of the out-of-place distance metric. Figure is reproduced from Cavnar and Trenkle (1994).

The feature selection component of the `TextCat` method can be understood as a local dimensionality reduction (Sebastiani 2002:Section 5.3). Local dimensionality reduction refers to dimensionality reduction by feature selection where a set of terms is selected for each category (in contrast to global dimensionality reduction, where terms are selected across all categories simultaneously). In the case of `TextCat`, the most frequent $M$ terms in each category are selected. The set of features selected is a function of both $M$ and the training data, and thus the number of features selected for a given $M$ varies with the training data.

The rank-order classifier of `TextCat` can in turn be understood as an implementation of a nearest-prototype classifier (aka Rocchio method (Rocchio 1971)). After feature selection has determined the closed set of features that will be used, each training document can be represented as a feature vector $V$, where $|V|$ is the num-

ber of features in the closed set, and each element of $V$ is the frequency count of a particular feature. The prototyping function is thus a simple sum across the feature vectors of all the training documents for a particular language (Figure 4.2), and the distance metric is the aforementioned out-of-place rank-order statistic (Figure 4.3).

For purposes of this comparison, we use the implementation of `TextCat` training and classification provided by `libtextcat` (Scheelen 2003). This implementation is written in pure C, and is optimized for both speed and memory efficiency. `libtextcat` is intended to be used as a library, and we access it via the Python binding `pylibtextcat`.[2] For generation of models, we use the `createfp` tool provided with `libtextcat`, with a minor modification to allow variation of the parameter $M$ (the number of features selected per language).

### 4.1.2 Linguini

`Linguini` (Prager 1999a; Prager 1999b) is a LangID algorithm using a vector-space model that is based on classical models for information retrieval. Prager (1999a) describes the model and provides an evaluation of it over 13 European languages, using a collection of data gathered from the (now-defunct) Human Languages Page,[3] including some synthetic data generated on the basis of word frequency statistics. Prager (1999a) also describes an extension to the method allowing for the detection of multilingual documents, which we separately evaluate in Chapter 6. Prager (1999b) is an extended version of Prager (1999a), which includes derivation of parameters for the detection of trilingual documents, extending the parameters for bilingual documents

[2] `https://launchpad.net/pylibtextcat`

[3] `http://www.june29.com/HLP`

given in Prager (1999a).

Prager (1999a) reports that the system described is internally used in an IBM product. To our knowledge, no implementation of the system has been made publicly available. Based on the description given by Prager (1999a), we created an implementation of the system, including the bilingual and trilingual extensions. The full system and training tools have been made available online under an open source license.[4]

The core of the `Linguini` system is a vector-space model. In a vector-space model, instances (in our case, documents), are interpreted as vectors in a multi-dimensional feature space. In vector-space models applied to text, each dimension usually corresponds to a particular word, and the magnitude in each dimension is a function of the frequency with which the word occurs in the document. The model used by `Linguini` is slightly different, in that in addition to words, there are also dimensions that represent the frequency of particular sequences of letters (also known as character $n$-grams, see Figure 2.1). Prager (1999a) considers letter sequences of length 2 – 5, not spanning words but possibly including word-ending spaces.

In `Linguini`, each entry in a document's feature vector is normalized by the number of languages in the training data that the feature appears in. This is very similar to to Term Frequency – Inverse Document Frequency (TF – IDF) models commonly used in vector space models for information retrieval, with the difference that instead of inverse document frequency, the term frequency in `Linguini` is normalized by inverse language frequency instead.

[4] `https://github.com/saffsd/linguini.py`

In the vector-space model used by `Linguini`, languages are also represented as vectors in the same vector-space as documents. The vector that represents a language can be found by taking the average in each dimension of all the vectors representing documents in that language in the training data. Classification of an unlabeled document is carried out by mapping the unlabeled document onto its corresponding vector, and then labeling the document with the language that is "closest" to the vector. In `Linguini` (and commonly in vector-space models in general), "closeness" is measured in angular terms using cosine similarity.

$$\frac{\sum^t N_{D,t} \cdot N_{C,t}}{\sqrt{\sum^t N_{D,t}^2}\sqrt{\sum^t N_{C,t}^2}} \tag{4.1}$$

Cosine similarity (Equation 4.1) is a function of two vectors, and gives a measure of how large the angle between the vectors is. Values of the function range between 0 and 1, with 0 indicating that the vectors are orthogonal and 1 indicating that the vectors are parallel. To classify an unlabeled document, the cosine similarity is calculated between the document vector $N_{D,t}$ and each language vector $N_{C,t}$, and the language vector with the highest similarity determines the language assigned to the unlabeled document.

Prager (1999a) evaluates the accuracy of the vector-space model using each order of $n$-grams independently. The representation reported to be optimal by Prager (1999a) is character 4-grams combined with words of any length; this is the representation that we will use for comparison in this thesis. Prager (1999a) applied a simple feature selection based on term frequency. Terms with occurrence count $m < n \cdot k$ were rejected, where $m$ is the number of times the term occurred in the training data, $n$ is the number of languages in which the term occurred and $k$ is a parameter to

control the overall number of terms selected. In Prager (1999a) the value of $k$ is reported to be optimal in the region 0.3 to 0.5. We examine this parameter in more detail in Section 4.2.2.

### 4.1.3 `LangDetect`

`LangDetect` implements a naive Bayes classifier. Documents and languages are represented as distributions over character $n$-grams, which similar to the representation used by `Linguini` except that `LangDetect` does not consider whole words. As an off-the-shelf classifier, `LangDetect` is distributed with a pre-trained model built using data from Wikipedia, but it can also be re-trained with user-supplied data. To improve accuracy off-the-shelf, `LangDetect` implements an additional set of normalization heuristics.

Naive Bayes classification is a family of probabilistic classification techniques. McCallum and Nigam (1998) describe two alternative formulations, and the formulation used by `LangDetect` is referred to by McCallum and Nigam (1998) as multinomial naive Bayes. The crux of the method is to compute the probability $P(C_i|D)$ that an instance $D$ to be classified belongs to a class $C_i$ from a given closed set $C$. The instance to be classified $D$ consists of a vector of $n$ features $x_1 \cdots x_m$, where each element of the vector corresponds to the frequency of a particular "event". In the context of naives Bayes classification applied to text, each "event" is the occurrence of a word in a document; the feature vector is thus simply a vector of frequencies of each word. In the LangID context, `LangDetect` follows a similar intuition, except that the "events" are the occurrence of character $n$-grams rather than words.

$$c = \text{argmax}_{C_i \in C} P(C_i|D) \tag{4.2}$$

$$= \text{argmax}_{C_i \in C} \frac{P(D|C_i)P(C_i)}{P(D)} \tag{4.3}$$

$$= \text{argmax}_{C_i \in C} P(D|C_i)P(C_i) \tag{4.4}$$

The classification $c$ of document $D$ is thus given by Equation 4.2. Bayes' theorem allows us to re-express Equation 4.2 as Equation 4.3. Since $P(D)$ is independent of $C_i$, this simplifies to Equation 4.4. To classify a document, we need to estimate $P(D|C_i)$ and $P(C_i)$. $P(C_i)$ is relatively straightforward. On the basis of our training data, we can estimate $P(C_i)$ using the maximum likelihood estimator, which is simply the relative frequency of classes in the training data.

The multinomial model is a generative model. Each class is modeled as a distribution over an event space corresponding to individual tokens. The naive independence assumption characteristic of naive Bayesian methods is to assume that the probability of generating a token is conditionally independent of the probability of generating any other token, given the class they are generated from.

$$P(D|C_j) = (\sum_{i=1}^{n} N_{D,t_i})! \prod_{i=1}^{n} \frac{P(t_i|C_j)^{N_{D,t_i}}}{N_{D,t_i}!} \tag{4.5}$$

Equation 4.5 expresses the probability of generating an instance $D$ as the probability of generating $x_1 \cdots x_m$, where $N_{D,t_i}$ is the frequency with which term $t_i$ occurs in $D$. The two factorial terms deal with permutations of tokens. Since $x_1 \cdots x_m$, are frequency counts of individual tokens, their sum is the length of the document (i.e.

the total number of tokens). For any sequence of $K$ items there are $K!$ possible permutations for ordering all the items. Under our "bag" model of text, we ignore the relative ordering of tokens. This implies that any ordering of the same set of tokens should have the same probability, and this is implemented through the two factorial terms. The first term accounts for all permutations of the entire set of tokens, and the second term accounts for all permutations of each type of token.

$P(t|C_j)$ is the generative model of class $C_j$. It is the probability of seeing a given term $t$ in class $C_j$. We estimate this via a maximum likelihood estimate on the basis of our training set, which consists of $k$ labeled instances $D_i \cdots D_k$.

$$P(t|C_j) = \frac{\sum_k^{|D|} N_{k,t} P(C_j|D_k)}{\sum_i^n \sum_k^{|D|} N_{k,t_i} P(C_j|D_k)} \tag{4.6}$$

Equation 4.6 gives the formula for $P(t|C_j)$, where $N_{k,t_i}$ is the count of term $t_i$ in $D_k$. This estimate poses another practical problem: If we have never observed a term in a given class in the training set, an instance containing that term will be assigned 0 probability for that class. Since we do not know every possible instance of each class in advance, we must build a residual probability of seeing any term into our model of the class. We do this via Laplacian smoothing, as shown in Equation 4.7.

$$P(t|C_j) = \frac{1 + \sum_k^{|D|} N_{D_k,t} P(C_j|D_k)}{n + \sum_i^n \sum_k^{|D|} N_{D_k,t_i} P(C_j|D_k)} \tag{4.7}$$

To reduce the total computation required, `LangDetect` makes use of the typical Bayesian model property that the posterior can be updated incrementally as each new observation is added, making use of the fact that given a document $D$ consisting

of tokens $X_1 \cdots X_{|D|}$, for any class $C_j$:

$$P(C_j|X_{m+1}, X_m, \cdots, X_1) \propto P(C_j|X_m, \cdots, X_1) \cdot P(X_{m+1}|C_j)$$

`LangDetect` thus introduces an "early-termination" technique, whereby the rest of the document is ignored if $P(C_j|X_m, \cdots)$ exceeds a threshold (reported by Nakatani (2010b) to be 0.99999).

Differently to `TextCat` and `Linguini`, `LangDetect` models the relative frequency of Unicode codepoint $n$-grams rather than byte $n$-grams. `LangDetect` documentation states that all documents are assumed to be encoded in UTF8, and does not specify what happens if this assumption is violated. Inspection of the source code shows that handling of encoding is deferred to the underlying Java libraries. The Java documentation does not specify what happens if a document cannot be decoded under a specified encoding, but based on our experiments it appears to silently ignore errors and produce a stream of codepoints from any stream of bytes, even if the stream of codepoints may not be coherent to a human reader. Hence, this transformation from bytes to codepoints under an incorrectly-specified encoding is still deterministic if perhaps slightly lossy, meaning that `LangDetect` still produces a working classifier even if applied to documents that are not UTF8 encoded.

Nakatani (2010b) reports that the naive Bayes classifier applied to byte $n$-grams only attains 90% precision (the $n$-gram order is not reported), and attributes this to bias and noise in the training and test corpora. In order to address this, a series of heuristics are introduced. Firstly, input text in the Chinese-Japanese-Korean space is clustered by k-means, such that each unique character is mapped into a cluster with other similar-feature characters. Nakatani (2010b) reports obtaining 130 such

clusters. Thereafter, a number of other heuristics are applied:

- Numeric figures, symbols, URLs and mail addresses are removed
- All Latin-characters are removed from non-Latin text if the rate is less than 20%
- Acronyms, names and place names are removed
- Words written entirely in capitals are removed

Nakatani (2010b) does not give further details of how these normalizations are carried out. The final result reported is an accuracy of 99.9% across 49 languages, using training data from Wikipedia and evaluated on 200 news articles per language.

## 4.2 Parameter Tuning

We first consider the "in-domain" evaluation setting typical of LangID research to date, where training and test data are disjoint partitions of a single dataset. This corresponds to a standard machine learning approach to implementing and evaluating LangID. In this section, we examine the parameters to be tuned, and empirically investigate their effect on classification accuracy across each of our datasets. `TextCat` and `Linguini` each have a tunable parameter that controls the number of features selected, whereas `LangDetect` does not have any user-tunable parameters in the training of the model.

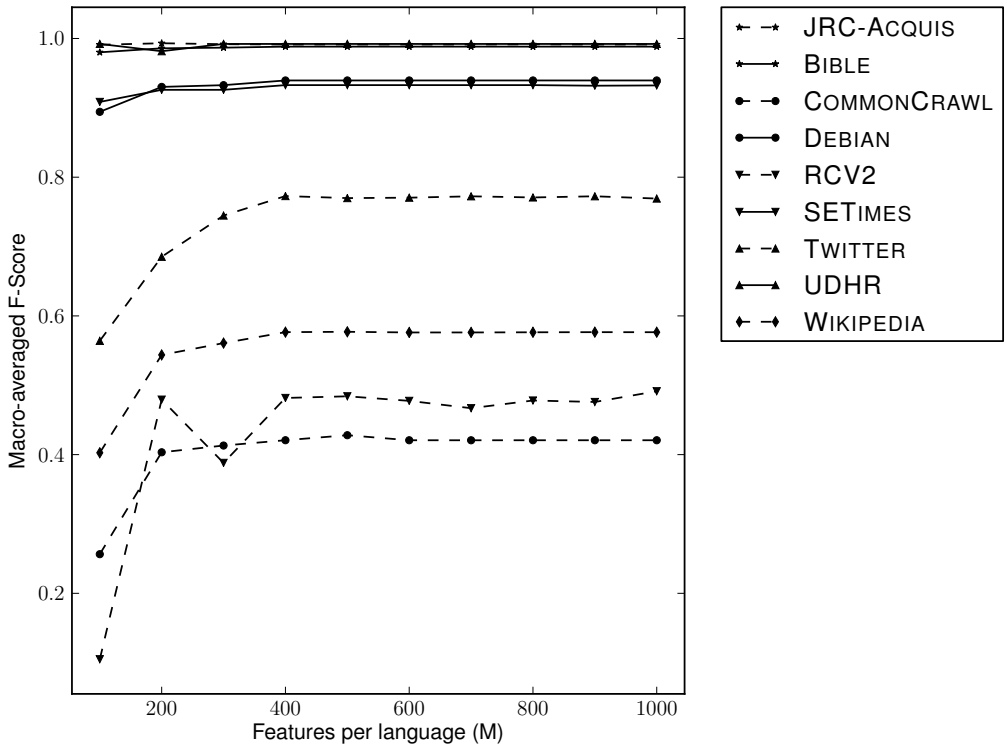



Figure 4.4: Parameter tuning for TextCat.

### 4.2.1 TextCat

In `TextCat`, the only parameter to be tuned is $M$, the number of features selected per language. Cavnar and Trenkle (1994) report that they found 400 to be the optimal value for $M$, and this is the value hard-coded into the model-generation tool supplied with `libtextcat`. We modified the tool to allow the value $M$ to be varied, and investigated its effect on an in-domain evaluation across each of our datasets. The result across all of our datasets is presented in Figure 4.4. We observe that consistent with the findings of Cavnar and Trenkle (1994), adding features beyond the top 400 by document frequency does not improve the accuracy of LangID.

One basic conclusion we can draw from Figure 4.4 is that the accuracy of `TextCat`

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.48 | 0.53 | 0.50 | 0.40 |
| **es** | | 0.47 | 0.56 | 0.36 |
| **fr** | | | 0.51 | 0.37 |
| **it** | | | | 0.37 |

JRC-Acquis

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.34 | 0.36 | 0.34 | 0.35 |
| **es** | | 0.40 | 0.46 | 0.31 |
| **fr** | | | 0.42 | 0.30 |
| **it** | | | | 0.32 |

Bible

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.84 | 0.78 | 0.86 | 0.77 |
| **es** | | 0.77 | 0.84 | 0.78 |
| **fr** | | | 0.79 | 0.73 |
| **it** | | | | 0.76 |

CommonCrawl

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.49 | 0.53 | 0.51 | 0.47 |
| **es** | | 0.50 | 0.58 | 0.40 |
| **fr** | | | 0.51 | 0.41 |
| **it** | | | | 0.41 |

Debian

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | – | – | – | – |
| **es** | | 0.81 | 0.81 | 0.79 |
| **fr** | | | 0.83 | 0.80 |
| **it** | | | | 0.86 |

RCV2

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.48 | 0.52 | 0.50 | 0.47 |
| **es** | | 0.55 | 0.62 | 0.46 |
| **fr** | | | 0.55 | 0.47 |
| **it** | | | | 0.47 |

Twitter

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.41 | 0.44 | 0.37 | 0.28 |
| **es** | | 0.43 | 0.41 | 0.25 |
| **fr** | | | 0.37 | 0.25 |
| **it** | | | | 0.26 |

UDHR

| | es | fr | it | sv |
|---|---|---|---|---|
| **en** | 0.53 | 0.54 | 0.52 | 0.46 |
| **es** | | 0.59 | 0.65 | 0.49 |
| **fr** | | | 0.59 | 0.49 |
| **it** | | | | 0.48 |

Wikipedia

Table 4.2: Proportion of $n$-grams shared between languages in a `TextCat` model selecting the most frequent 1000 character $n$-grams per-language.

varies widely according to the source of text. In JRC-Acquis, Bible and UDHR, the accuracy is near-perfect. However, in noisier domains, the accuracy is severely affected, such as in Wikipedia, where the macro-averaged F-score is about 0.58. This can partly be explained due to the noise in the domain affecting the model due to the term-frequency based feature selection employed by `TextCat`. In CommonCrawl for example, documents contain HTML markup, and the markup will account for a large proportion of the character $n$-grams in a document. Hence, these 'noise' $n$-grams will have high frequency in most languages and will be included in the final model, but will not have good discriminating power for any given language. Table 4.2 lists

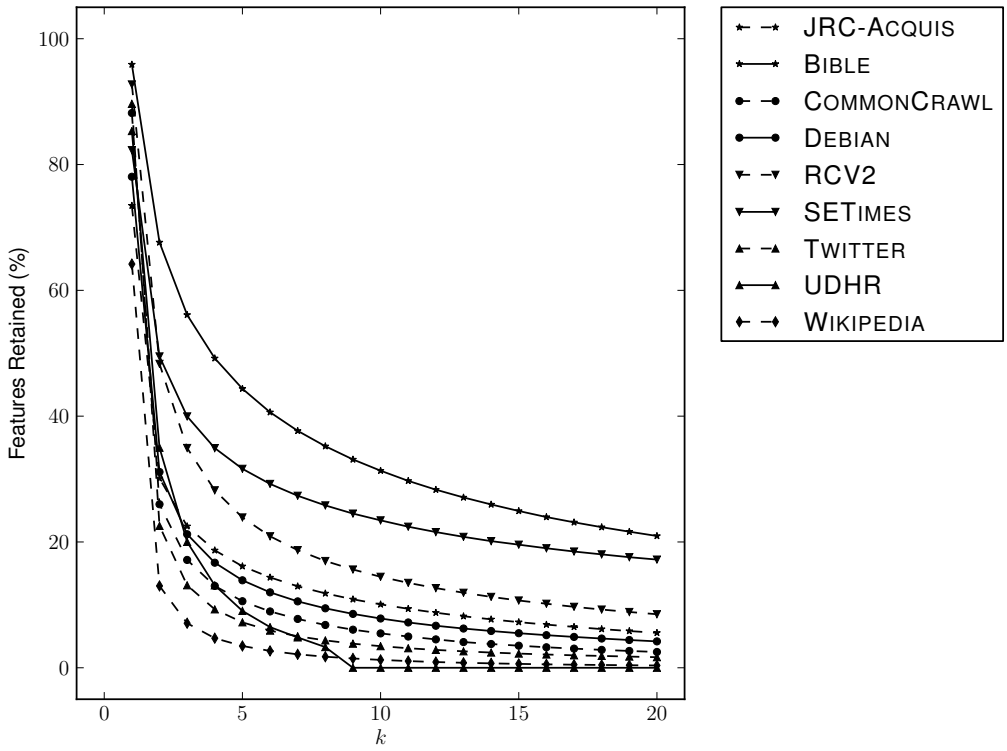



Figure 4.5: Parameter tuning for Linguini.

the proportion of character $n$-grams shared between given pairs of languages when each language is modeled using the top 1000 character $n$-grams by frequency. In text sources where accuracy is high, the overlap tends to be lower. On the other hand, where overlap is high (CommonCrawl and RCV2), the accuracy tends to be lower. This is because the frequency-based selection is not always reliable in selecting features that are indicative of a particular language, an issue that we will explore and resolve in Chapter 5.

| **Dataset** | **# Features** | $k = 10$ | | $k = 20$ | |
|---|---|---|---|---|---|
| | | # | % | # | % |
| JRC-Acquis | 737754 | 74371 | 10 | 40999 | 5.5 |
| Bible | 486259 | 152347 | 31 | 101879 | 21 |
| CommonCrawl | 4713952 | 257604 | 5.5 | 118447 | 2.5 |
| Debian | 1298254 | 101675 | 7.8 | 54471 | 4.2 |
| RCV2 | 383079 | 55411 | 14 | 32604 | 8.5 |
| SETimes | 212436 | 49781 | 23 | 36538 | 17 |
| Twitter | 422423 | 14510 | 3.4 | 7097 | 1.6 |
| UDHR | 103902 | 0 | 0.0 | 0 | 0.0 |
| Wikipedia | 1414614 | 17723 | 1.25 | 5158 | 0.36 |

Table 4.3: Number of features selected for `Linguini` at different values of $k$.

### 4.2.2 Linguini

`Linguini` uses a parameter $k$ to control the number of features selected. Prager (1999a) gives two slightly conflicting definitions of this parameter: (1) "If feature $i$ occurred $m_i$ times in a language training set, the value we stored was the integral part of $k{\cdot}m_i/n_i$", where $n_i$ is the number of languages the feature $i$ occurs in. (2) "a word would not be stored if its occurrence count $m_i < n_i{\cdot}k$". Simple algebra shows that the two definitions are consistent if $\frac{1}{k}$ is substituted for $k$ in either definition. In our implementation, we use the second definition for $k$: specifically, features are kept only if $m_i \geq n_i{\cdot}k$.

Prager (1999a) reports that values of $k$ from 0.1 to 10 were tested, and that values in the region of 0.3 to 0.5 worked best. It is not clear if the value of $k$ should be interpreted in the context of the first or second definition, and it is not possible to infer from the reported results as Prager (1999a) does not report the number of features selected or the quantity of data available for each language. This latter value is important as it affects the effect that the value $k$ has. Since the selection is based

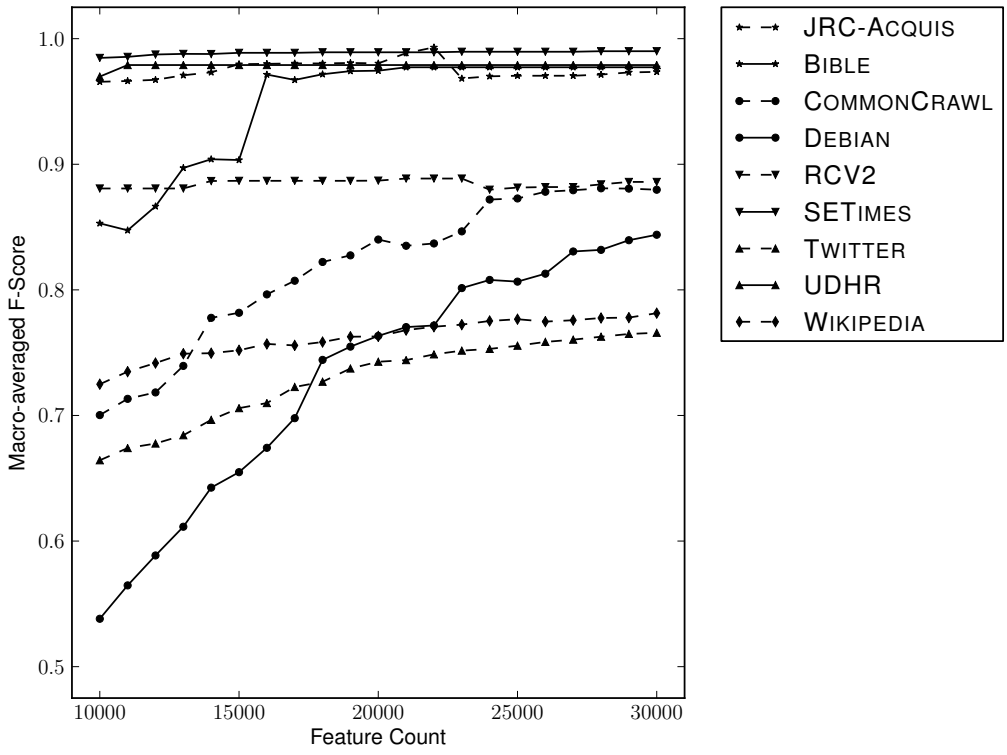



Figure 4.6: Effect of feature count on `Linguini` accuracy.

on $m_i$, the frequency with which a feature occurs, it will increase with data quantity. Since we use a different dataset from Prager (1999a), the results for the same value of $k$ are not comparable.

We begin by investigating the effect of the parameter $k$. Figure 4.5 shows the relationship between $k$ and the proportion of features retained for each dataset. We observe that the proportion of features retained for the same value of $k$ varies per-dataset. This is further complicated by the datasets being of different sizes, resulting in the absolute number of features retained varying greatly (Table 4.3).

Fundamentally, the role of $k$ is to control the number of features – which then corresponds to the number of dimensions in the vector space of the `Linguini` model.

Increasing the value of $k$ simply has the effect of decreasing the dimensionality of the model. To allow for better comparison between the models resulting from different datasets, we drop the notion of the fixed threshold $k$ and instead simply retain the top-$N$ features, ranked by the value $\frac{m_i}{n_i}$. We test $10000 \leq N \leq 30000$; the result is presented in Figure 4.6. For some datasets, the difference between $N = 10000$ and $N = 30000$ is minimal, such as for SETIMES and for RCV2. For other datasets, there is a continued increase as more features are added, such as for DEBIAN. One factor that this analysis does not take into account is the feature selection method described by Prager (1999a) is a global feature selection (Sebastiani 2002), i.e. features are scored across all classes simultaneously. Because the feature selection is based on term frequency, and because the amount of data available for each language is not consistent within datasets, there will be a natural tendency for features strongly associated with highly-represented languages to score highly on the metric.

For further comparisons in this chapter, we select $N = 25,000$ features as the optimal value, as it appears that for most datasets the increase in accuracy thereafter is minimal.

## 4.3 In-domain Comparison

After tuning the parameters for each tool using the DEV portion of each dataset, we now use the best parameters for each tool to classify the TST portion of each dataset, and use the results as the basis for a cross-tool comparison. For `TextCat`, we set the parameter $M$ to 400, and for `Linguini`, we select the top 25,000 features. The results of this comparison are summarized in Table 4.4. From these results, we

| Dataset | TextCat | | Linguini | | LangDetect | |
|---|---|---|---|---|---|---|
| | **M** | $\mu$ | **M** | $\mu$ | **M** | $\mu$ |
| JRC-Acquis | 0.991 | 0.990 | 0.977 | 0.977 | 0.994 | 0.994 |
| Bible | 0.990 | 0.985 | 0.985 | 0.984 | 0.991 | 0.993 |
| CommonCrawl | 0.429 | 0.387 | 0.881 | 0.887 | 0.825 | 0.836 |
| Debian | 0.926 | 0.948 | 0.788 | 0.874 | 0.945 | 0.971 |
| RCV2 | 0.453 | 0.557 | 0.887 | 0.929 | 0.926 | 0.939 |
| SETimes | 0.933 | 0.933 | 0.987 | 0.988 | 0.993 | 0.993 |
| Twitter | 0.765 | 0.807 | 0.753 | 0.802 | 0.819 | 0.870 |
| UDHR | 0.981 | 0.984 | 0.980 | 0.984 | 0.982 | 0.988 |
| Wikipedia | 0.555 | 0.521 | 0.759 | 0.729 | 0.628 | 0.618 |

Table 4.4: In-domain comparison of systems. **M** indicates macro-averaged F-score, and $\mu$ indicates micro-averaged F-score. $N = 25k$ for `Linguini` and $M = 400$ for `TextCat`.

observe that in some datasets, all the tools are able to attain a high degree of accuracy (e.g. JRC-Acquis, UDHR, Bible). It is worth noting that these datasets are quite similar to those commonly used in LangID research. JRC-Acquis covers a relatively small number of Western European languages, with long and well-curated texts in each language without any domain-specific document markup, similar to the dataset used by Cavnar and Trenkle (1994). Data from the UDHR was used by Yamaguchi and Tanaka-Ishii (2012), and bible texts have been used by Hammarström (2007) and Brown (2012).

Certain datasets reveal particular weakness in specific tools. For `TextCat`, it performs particularly poorly on CommonCrawl and RCV2. Notably, both these datasets are heavy in HTML/XML markup. As we observed in Table 4.2, poor accuracy for `TextCat` correlates with datasets where there is a substantial overlap between the highest-frequency character $n$-grams for each language. This overlap

is due to the same character $n$-grams from the markup being common in the data for each language. Frequency-based selection of features thus results in language representations that are not distinctive for each language, leading to reduced accuracy. `Linguini` is not affected by this as the inverse-language-frequency term serves to weight down the markup-related features, which occur in all the languages; we thus see that `Linguini` substantially outperforms `TextCat` on CommonCrawl and RCV2.

Relative to the other two systems, `Linguini` performs poorly on Debian. We note that in Figure 4.6, the accuracy on Debian continues to increase beyond $N = 25k$, so the relatively poor performance in this case is due to the number of features selected being insufficient to reliably distinguish all the languages present in Debian. This is further emphasized by the relatively large gap between the macro-averaged and micro-averaged F-score, which indicates that accuracy is particularly poor for a relatively small proportion of the lower-density languages.

`LangDetect` generally performs well across all datasets, with the exception of Wikipedia. This is likely due in part to a tendency to mis-label documents as English – precision on English on the Wikipedia classifier is only 16%. This may be due to a number of factors, such as English-language inclusions being more common in Wikipedia than other datasets, either due to inclusion of English-language content, or through the use of English words in the structural markup of the raw MediaWiki-format documents.

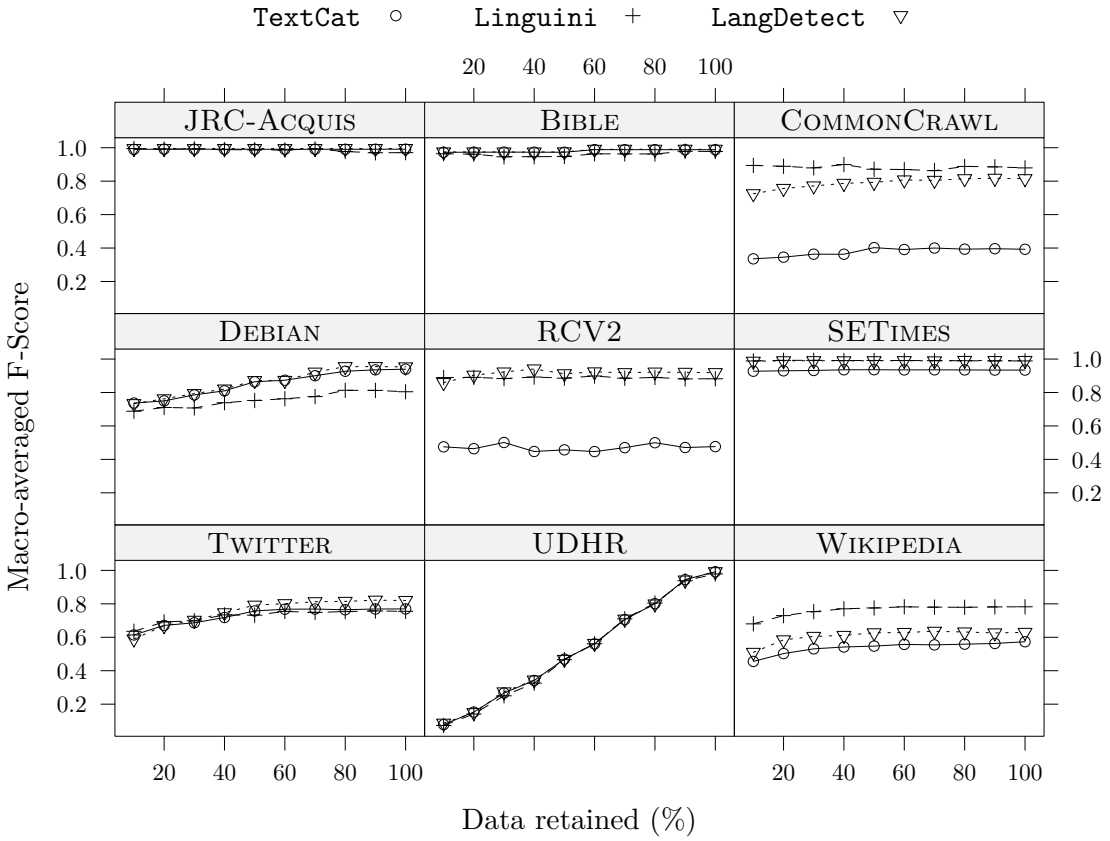



Figure 4.7: Learning curve for each combination of system and dataset.

### 4.3.1 Learning Curves

We investigate how the accuracy of each system is affected by varying the amount of training data provided. In this instance, our primary focus is to compare the performance of the different systems on the same dataset (or subset thereof), rather than to compare the performance of the same system on different datasets. We thus choose to break the TRN portion of each dataset into 10 approximately equal-sized partitions, stratified by language and randomized. This makes cross-dataset comparisons difficult as each dataset has a different total amount of data. However,

the results presented here in are still useful to justify the downsampling of data carried out in Section 4.4.2.

For `TextCat` and `Linguini` we use the previously-mentioned best parameters (Section 4.2). We plot a learning curve of percentage of training data used vs macro-averaged F-score; this is presented in Figure 4.7. We observe that for a few of the datasets (JRC-Acquis, Bible, SETimes), reducing the training data to 10% of the total available has no discernible effect on accuracy, which is virtually identical for all 3 systems. We also observe that `TextCat` performs consistently worse than the other systems on Wikipedia, RCV2 and CommonCrawl, due to the noise-related issues we discussed in Section 4.2.1. We also note that, consistent with Section 4.2.2, `Linguini` seems to be particularly weak on the Debian dataset, though this can be attributed to an insufficient number of features to cover all the languages present in Debian. The biggest improvement with increased data is seen on UDHR, followed by Twitter. This is not particularly surprising, since these are smallest datasets in terms of bytes of data available. Overall, we can draw two main conclusions from this result: (1) given the same training and test data, the three systems are generally very closely matched, and (2) the amount of training data is generally not a limiting factor for in-domain evaluation using our full training data.

## 4.4 Cross-domain Evaluation

So far, we have presented results that show that LangID tools can generally attain a high level of accuracy when the training data is drawn from the same pool of documents as the test data, consistent with similar high-accuracy results reported in

the literature (Cavnar and Trenkle 1994; Cowie *et al.* 1999; Prager 1999a; Poutsma 2002; Kruengkrai *et al.* 2005; Takçi and Ekinci 2012). However, in this thesis we are specifically interested in LangID for data from sources where no labeled training data is available. In this section, we quantify how existing systems perform when trained using training data that is from a different source to the target data. We formulate our empirical investigation in terms of two specific experiments: (1) single-source, where we use training data from one dataset to build a model, and evaluate the accuracy on data from a different dataset; and (2) all-source, where we hold out one dataset at a time as our "test" dataset, and use the union of all other datasets as a source for training data.

### 4.4.1 Single-source

In this setting, we use the TRN partition of each dataset to train each classifier, and apply the classifier to the TST partition of every dataset. The trained classifiers are exactly the same as those used in Section 4.3; the difference in this section is that we apply the classifiers to the TST partitions of each dataset, rather than only the TST partition of the same dataset the TRN data is from. One complication in this evaluation is that the set of languages covered by each dataset is different, making it difficult to compare the performance of the same combination of classifier and training data across the TST partition of different datasets. To allow comparability across different datasets, we present results in terms of macro-average performance over the subset of languages present in all the datasets (i.e. the intersection of the language set). We exclude SETIMES from this evaluation, leaving us with five languages that

Test Dataset

| | JRC-Acquis | Bible | CommonCrawl | Debian | RCV2 | Twitter | UDHR | Wikipedia |
|---|---|---|---|---|---|---|---|---|
| *In-domain* | 0.986 | 0.998 | 0.239 | 0.987 | 0.391 | 0.776 | 1.000 | 0.599 |
| JRC-Acquis | — | −0.216 | +0.029 | −0.134 | +0.086 | −0.043 | −0.616 | −0.093 |
| Bible | −0.001 | — | +0.045 | −0.106 | +0.049 | −0.082 | −0.310 | −0.076 |
| CommonCrawl | −0.616 | −0.668 | — | −0.692 | −0.316 | −0.703 | −0.800 | −0.421 |
| Debian | −0.006 | −0.053 | +0.004 | — | +0.016 | −0.125 | −0.253 | +0.091 |
| RCV2 | −0.955 | −0.974 | −0.201 | −0.935 | — | −0.662 | −0.989 | −0.586 |
| Twitter | −0.002 | −0.244 | +0.065 | −0.131 | +0.052 | — | −0.560 | −0.099 |
| UDHR | −0.007 | −0.033 | −0.018 | −0.068 | +0.025 | −0.220 | — | +0.073 |
| Wikipedia | −0.152 | −0.220 | −0.091 | −0.216 | −0.055 | −0.229 | −0.387 | — |

`TextCat`

Test Dataset

| | JRC-Acquis | Bible | CommonCrawl | Debian | RCV2 | Twitter | UDHR | Wikipedia |
|---|---|---|---|---|---|---|---|---|
| *In-domain* | 0.992 | 1.000 | 0.813 | 0.997 | 0.895 | 0.841 | 1.000 | 0.764 |
| JRC-Acquis | — | −0.322 | −0.456 | −0.182 | −0.231 | −0.153 | −0.672 | −0.488 |
| Bible | −0.012 | — | −0.441 | −0.112 | −0.253 | −0.122 | −0.363 | −0.473 |
| CommonCrawl | −0.002 | −0.133 | — | −0.122 | −0.073 | −0.055 | −0.400 | −0.386 |
| Debian | −0.022 | −0.015 | −0.515 | — | −0.239 | −0.082 | −0.210 | −0.239 |
| RCV2 | −0.314 | −0.428 | −0.361 | −0.279 | — | −0.296 | −0.590 | −0.464 |
| Twitter | −0.035 | −0.257 | −0.288 | −0.159 | −0.228 | — | −0.560 | −0.529 |
| UDHR | −0.042 | −0.015 | −0.482 | −0.064 | −0.346 | −0.163 | — | −0.237 |
| Wikipedia | −0.001 | −0.009 | −0.408 | −0.037 | −0.167 | −0.105 | −0.233 | — |

`LangDetect`

Test Dataset

| | JRC-Acquis | Bible | CommonCrawl | Debian | RCV2 | Twitter | UDHR | Wikipedia |
|---|---|---|---|---|---|---|---|---|
| *In-domain* | 0.973 | 0.947 | 0.905 | 0.982 | 0.956 | 0.793 | 1.000 | 0.754 |
| JRC-Acquis | — | −0.279 | −0.348 | −0.283 | −0.463 | −0.266 | −0.616 | −0.434 |
| Bible | −0.089 | — | −0.349 | −0.194 | −0.305 | −0.398 | −0.422 | −0.277 |
| CommonCrawl | −0.279 | −0.386 | — | −0.303 | −0.456 | −0.481 | −0.420 | −0.360 |
| Debian | −0.062 | −0.365 | +0.020 | — | −0.055 | −0.279 | −0.417 | −0.163 |
| RCV2 | −0.378 | −0.316 | −0.473 | −0.381 | — | −0.315 | −0.656 | −0.517 |
| Twitter | −0.098 | −0.390 | −0.034 | −0.172 | −0.041 | — | −0.550 | −0.341 |
| UDHR | −0.006 | −0.056 | −0.782 | −0.128 | −0.555 | −0.496 | — | −0.173 |
| Wikipedia | −0.437 | −0.412 | −0.549 | −0.422 | −0.670 | −0.723 | −0.800 | — |

`Linguini`

Table 4.5: Cross-domain single dataset evaluation.

are present in every other dataset: Swedish (sv), Italian (it), French (fr), Spanish (es) and Danish (da). Thus, all values reported in this section are the macro-average across these 5 languages. Note that the full set of languages present in each dataset was used for training, so it is possible for a test document in one of these 5 languages to receive a label outside the 5 languages, which would count as a false negative for that language. We also use the full set of documents from each TST partition. Many of the documents will thus have a correct label outside this restricted 5-language set.

If a classifier labels a document outside this set with a language from within the set, this counts as a false positive for the language.

The results of this experiment are summarized in Table 4.5. The first row for each classifier (labeled *In-domain*) reports the *absolute* macro-averaged F-score across the 5 above-mentioned languages for the case where the training and test data come from the same dataset. Note that these values are different from Table 4.4, because Table 4.4 reports the average score across all languages in each dataset, whereas the *In-domain* row of Table 4.5 reports average score in the 5-language subset. The remaining rows for each classifier report cases where the training and test data come from different datasets, and the values are the *difference* to the *In-domain* value in the same column. This allows for easy interpretation of the results in terms of the effect of choosing a different source of training documents for a given source of test documents.

One trend that we observe is that documents for certain sources are "easier" to classify than others, in that classifiers consistently do well regardless of training data used. For example, most dataset-classifier combinations produce a system that does well for classifying documents from the JRC-Acquis dataset. A notable exception is the use of RCV2 data, which results in consistently poor performance. Indeed, RCV2 is a poor choice as a source of training data except for in-domain LangID. The best result on RCV2 is obtained by `Linguini` trained on RCV2 data. Interestingly, training `Linguini` on Twitter or Debian data produces a classifier that does well on RCV2, but the property is not commutative: a `Linguini` model trained on RCV2 data is much worse than an in-domain model of either Twitter or Debian. This

asymmetry is a common theme: UDHR data produces good results on JRC-Acquis and Bible across all systems, but JRC-Acquis and Bible consistently produce poor results on UDHR. We can draw several conclusions from this experiment: (1) pairwise performance between data sources is not commutative, i.e. source X producing good results on source Y does not imply source Y will produce good results on source X; (2) all classifiers produce the best results when using in-domain training data (the slight exception being `TextCat` on CommonCrawl and RCV2, where in-domain results were poor to begin with, and are only marginally improved by cross-domain data); (3) certain text sources are uniformly "easy" for all classifiers when in-domain data is available (JRC-Acquis, Bible, UDHR), whereas others are consistently harder (e.g. Twitter and Wikipedia); and (4) there is no single combination of system and data source that is effective for all target domains. (4) is especially critical in the context of *generalized* LangID as it implies that using existing methods, training a model on a single source of training data is likely to produce a classifier with poor accuracy, because there is no single source of training data that performs uniformly well when applied to different test data. In the next section, we investigate the consequences of a simple solution to this problem, the use of training data from a combination of different sources, without taking into account the source of the data.

### 4.4.2 All-source

In the previous section, we quantified the effect on LangID accuracy of using training data from any single source other than the test domain. We found that there was no single combination of system and data source that produced a classifier

that was equally effective in all domains. However, it is likely that even if no labeled data is available in the target domain, labeled training data may be available from several other domains. In this section, we investigate whether the simple union of such training data is a suitable approach to tackling the cross-domain LangID problem.

We combine training data from all the datasets except a single held-out "test" dataset, treating each dataset as the "test" dataset in turn. We use the combined data to train each classifier, and then apply this to the test data from the held-out dataset. For the datasets that we have prepared for the purposes of this thesis, each language is present in at least two datasets. Hence, in this section, we present results that are macro-averaged over the full language set covered by the "test" dataset. These results are thus directly comparable to the in-domain results presented in Section 4.3.

Combining training data from multiple domains increases the amount of training data used to produce the model. As we saw in Section 4.3.1, for most datasets the quantity of training data is not a limiting factor, and the training data can be scaled back to 10% with minimal effect on accuracy. Thus, for this experiment we use the whole of Twitter-trn and UDHR-trn, but downsample each of JRC-Acquis, Bible, CommonCrawl, Debian, RCV2, SETimes and Wikipedia to 10% of the original data, randomized and stratified by language.

Table 4.6 summarizes the results of this experiment. In it, we present the in-domain result from Section 4.3 alongside a delta value for the cross-domain result. The in-domain result is the *absolute* macro-averaged F-score across all the languages in the target domain attained by a model trained using training data from only the target domain. The cross-domain result is the *relative* difference to the absolute score

| Dataset | TextCat | | Linguini | | LangDetect | |
|---|---|---|---|---|---|---|
| | In-domain | Cross-domain $\delta$ | In-domain | Cross-domain $\delta$ | In-domain | Cross-domain $\delta$ |
| JRC-Acquis | 0.991 | −0.373 | 0.977 | −0.176 | 0.994 | −0.080 |
| Bible | 0.990 | −0.383 | 0.985 | −0.378 | 0.991 | −0.225 |
| CommonCrawl | 0.429 | −0.239 | 0.881 | −0.201 | 0.826 | −0.601 |
| Debian | 0.926 | −0.285 | 0.788 | −0.258 | 0.945 | −0.198 |
| RCV2 | 0.453 | −0.085 | 0.887 | −0.466 | 0.926 | −0.133 |
| SETimes | 0.933 | −0.294 | 0.987 | −0.372 | 0.993 | −0.280 |
| Twitter | 0.765 | −0.202 | 0.753 | −0.353 | 0.819 | −0.181 |
| UDHR | 0.982 | −0.443 | 0.980 | −0.493 | 0.982 | −0.327 |
| Wikipedia | 0.555 | −0.091 | 0.759 | −0.476 | 0.628 | −0.167 |

Table 4.6: Comparison of in-domain and cross-domain results for each classifier and dataset combination.

obtained by a model trained using the union of training data from all domains *except* the target domain.

It is immediately apparent that utilizing the union of out-of-domain training data consistently produces a result that is inferior to utilizing in-domain training data. The size of the difference varies both with the target domain and the system used. Overall, it would appear that, of the three systems tested, `LangDetect` is the most resilient when trained on out-of-domain training data. However, the results obtained are still significantly below those obtained on in-domain training data. In this chapter, we have demonstrated that simply pooling training data from multiple datasets is not an effective way to train a generalized language identifier, regardless of the underlying LangID system used. In order to achieve generalized LangID, our method must take into account not just the difference between languages, but also the source of the training data used. In Chapter 5 we examine this issue more closely, investigating the root cause of the difference in accuracy of classifiers trained on in-domain data versus classifiers trained on out-of-domain data, and developing a strategy that mitigates the loss in accuracy when applying a language identifier to a source of text that is

different from that which it was trained on.

## 4.5 Chapter Summary

In this chapter, we provided a systematic comparison of three commonly used LangID systems. We discussed their theoretical similarities and differences, and then compared them empirically, using standardized training and test data to allow for direct comparison of the three systems. We found that in an in-domain evaluation setting, which is common practice in LangID research to date, all systems performed well using data from the sources commonly used for evaluating LangID systems, such as data from UDHR, Bible passages or European Government documents. We also found that `TextCat` performed poorly on data heavy in XML-markup, whereas the other two systems compared were relatively robust. Finally, we found that despite the use of in-domain training data, performance on data from Twitter and Wikipedia was relatively poor for all systems tested.

In contrast to in-domain evaluation, we also empirically evaluated the effect of using data from sources other than the target domain. We did this under two settings: (1) using data from a single other source, and (2) pooling data from multiple sources. In the former case, we found that no single source of training data performed well in all domains, and that pairwise performance between domains is asymmetric. In the latter case, we found that pooling data generally produces inferior results to using in-domain data, and that the penalty incurred varies by target domain and system used.

Overall, in this chapter we have demonstrated that while commonly-used LangID

tools can be very effective under certain circumstances replicating those frequently reported in the literature, none of the systems that we examined is effective as a generalized language identifier. In the next chapter, we will explore in more detail why this is the case, and will formulate solutions that will lead to better performance for generalized LangID.

# Chapter 5

# Document Representation for Generalized LangID

In the previous chapter, we examined three existing LangID systems, and empirically evaluated their accuracy in both in-domain and cross-domain contexts. We found that whereas all the classifiers performed well when the training data was drawn from the same source as the test data, all the classifiers also exhibited a decline in accuracy when the training and test data were drawn from different sources. In this chapter, we investigate the underlying causes of this decline, and develop a strategy to mitigate this loss in performance caused by lack of data in the target domain. In Section 5.1, we provide an analysis of the inner workings of the three systems that

This chapter is based on work previously published as:

Lui, Marco, and Timothy Baldwin. 2011. Cross-domain Feature Selection for Language Identification. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, 553 – 561, Chiang Mai, Thailand.

we discussed in Chapter 4, showing how the systems share common concepts in the representation of documents and languages. We then discuss how each system determines the most likely language for a document (Section 5.2), and relate this to supervised machine learning (Section 5.3). We then introduce the concept of homogeneity from corpus linguistics (Section 5.3.1), relating it to the assumptions made in the inductive learning hypothesis in machine learning. We empirically evaluate the homogeneity with respect to language of the datasets we describe in Chapter 3 under the modeling assumptions made by the systems described in Chapter 4, and show that the pre-conditions of supervised machine learning are not met in the cross-domain classification setting described in Section 4.4. We relate this problem to transfer learning (Section 5.3.3), and present a cross-domain feature selection methodology that improves the homogeneity of the document representation with respect to language across different datasets (Section 5.4). We then examine the properties of the learning algorithms used by the systems we discussed in Chapter 4, identifying the properties which make them suitable for LangID (Section 5.5.1). We apply each learning algorithm to our novel document representation (Section 5.5.2), and show that across all the algorithms we consider, our novel document representation improves the accuracy of cross-domain LangID (Section 5.5.4). Thereafter, we provide an error analysis (Section 5.6), in which we examine and discuss the reasons for which documents are misclassified in our experiments.

## 5.1 Document and Language Representation

We begin our discussion by more closely considering the concepts underpinning the design of the three LangID systems that we investigated in Chapter 4. We decompose each system into two distinct components: (1) document representation and (2) language determination. The former refers to the structure of the internal representation of documents in each classifier. For all systems, each document is first converted into a vector of numbers; in this section we discuss how each system does this, and the similarities and differences between each approach. Thereafter, the most likely language for the document is determined by some function of the internal representation. In Section 5.2, we compare and contrast the approaches used by each system.

`LangDetect` represents documents as a vector of frequency counts over Unicode codepoint $n$-grams (Nakatani 2010b). Before division into $n$-gram sequences, the input text is normalized by applying a variety of heuristic rules. These rules make heavy use of the Unicode metadata associated with the input codepoints, and include transformations such as whitespace tokenization (to exclude $n$-gram sequences across "word" boundaries, consistent with Cavnar and Trenkle (1994)), case folding, and "codeplane reduction", where codepoints from specific Unicode blocks such as punctuation as well as more language-specific blocks such as Japanese-language Hiragana/Katakana script are mapped to a single codepoint used to represent the entire block. Some block-specific normalization is also applied, such as the conversion of the Farsi *yeh* (ى) into the Arabic equivalent (ي). Thereafter, a standard division into overlapping codepoint unigrams, bigrams and trigrams is applied (see Figure 2.1),

and the total vocabulary of trigrams is trimmed, using rules based on minimum frequency, as well as elimination of "latin" trigrams from documents that are detected to have less than one third of words written in latin script. The rationale for this last rule is to eliminate minority latin-script inclusions found in training data.

The core representation of documents used by `Linguini` is also a vector of frequency counts. However, unlike `LangDetect`, `Linguini` does not make the assumption that the input stream has been decoded into Unicode codepoints, and instead the representation is frequency counts over raw byte sequences. Prager (1999a) investigated a number of options for the "tokenization" to reduce a document into byte sequences, the frequency of which would then make up the document representation. Options considered included byte $n$-grams for $2 \leq n \leq 5$, as well as "short words", i.e. whitespace-delimited sequences of letters of length less than or equal to 4, and simply "words", i.e. whitespace-delimited sequences of letters of any length. Based on the experiments conducted, Prager selected a combination of byte 4-grams and words of unrestricted length. In principle, this leads to a feature set of infinite cardinality. For ease of implementation, each feature is then scored by the product of its term frequency and inverse language frequency (ILF), and a threshold $k$ is introduced to prune features that fall below the threshold. In practice, this has the effect of selecting the $M$ highest-scoring features by this TF-ILF metric, where $M$ is a function of $k$ as well as the underlying dataset. For easier comparison across datasets, in this thesis we parametrize `Linguini` by $M$ directly.

The document representation used by `TextCat` is a ranklist of tokens present in a document, ordered by descending frequency. Like `LangDetect`, the tokens used are $n$-

| **Feature** | TH | ER | ON | ED | ING | AND |
|---|---|---|---|---|---|---|
| **Frequency** | 25 | 4 | 10 | 2 | 12 | 3 |
| **Rank** | 1 | 4 | 3 | 6 | 2 | 5 |

Table 5.1: Example of rank order statistics of a frequency vector. Note that the frequencies and ranks are for explanatory purposes only and do not reflect any real $n$-gram frequency statistics.

gram sequences of mixed $n$-gram order. In the case of `LangDetect`, $1 \leq n \leq 3$, whereas in `TextCat` the upper bound used is $n = 4$. Unlike `LangDetect` (but like `Linguini`), `TextCat` generates $n$-gram "tokens" on the basis of the byte representation of the document rather than the codepoint representation. `TextCat` also strips punctuation, and disallows $n$-gram sequences that span word boundaries.

Superficially, the `TextCat` representation may appear quite different from the frequency vectors used by `LangDetect` and `Linguini`. In practice however, it is quite straightforward to implement the `TextCat` representation as a frequency vector, because the underlying similarity computation utilized by `TextCat` is an ad-hoc rank order statistic over a finite vocabulary of byte sequences. We discuss this in more detail in the next section; for now it is sufficient to note that the rank order statistics can be easily computed from the frequency vector (Table 5.1).

## 5.2 Classification Algorithms

In the previous section, we discussed the document representation used by each of the systems we considered in Chapter 4. In this section, we shift our attention to the next step in the process of identifying the language that a document is written in:

| Feature | Category Rank | Document Rank | Out-Of-Place |
|---|---|---|---|
| TH | 1 | 1 | $\|1-1\| = 0$ |
| ER | 2 | 4 | $\|2-4\| = 2$ |
| ON | 3 | 3 | $\|3-3\| = 0$ |
| LE | 4 | NA | $no-match = max$ |
| ING | 5 | 2 | $\|5-2\| = 3$ |
| AND | 6 | 5 | $\|6-5\| = 1$ |
| ED | NA | 6 | $no-match = max$ |
| Total | – | – | 26 (assuming max=10) |

Table 5.2: Example of calculating the out of place metric. NA indicates that a feature is missing from a particular profile.

the classification algorithms. After converting a document into the respective representations described in Section 5.1, each system computes a score for the document with respect to each language that it knows about, and then returns the top-scoring language as the most likely language for the document to be written in. In this section, we examine the approach taken by each system in detail, in particular focusing on the common intuition that underpins the three approaches we examine.

`TextCat` selects a candidate language for a document by comparing the ranklist of features present in a document to pre-computed ranklists for each language, and selecting the most similar under the "out-of-place" metric described by Cavnar and Trenkle (1994). The "out-of-place" metric can be simply stated as the sum of the absolute differences in ranks over a closed set of features. More formally,

$$c = \text{argmin}_{C_i \in C} \sum^{t} |Rank(t, D) - Rank(t, C_i)| \tag{5.1}$$

where $Rank(t, X)$ is the ranking of term $t$ in $X$, where $X$ can be the $n$-gram profile of either a document or a category (i.e. language). An example of calculating the out-of-

place metric is given in Table 5.2, adapted from Figure 3 of Cavnar and Trenkle (1994) – which we reproduced as Figure 4.3. The ranklist for a language is computed by concatenating all the training documents for the given language into a single pseudo-document, and computing the ranking of features in the same way as for a normal document.

`Linguini` uses a vector-space model, and so the "distance" between a document and a given language is given by the cosine of the angle between the two. Smaller angles lead to a smaller cosine, so for classification purposes the language with the smallest cosine to the document is selected:

$$c = \text{argmax}_{C_i \in C}\left(\frac{\sum^t N_{D,t} \cdot N_{C,t}}{\sqrt{\sum^t N_{D,t}^2}\sqrt{\sum^t N_{C,t}^2}}\right) \tag{5.2}$$

The representation of a language is the centroid of the vectors for all the training documents in the language, which is computed as the arithmetic mean of all the vectors for the language.

The Bayesian classifier used by `LangDetect` was described in more detail in Section 4.1.3. In summary, the score for each class is determined by an inner product between the document vector and a vector for each class.

$$c = \text{argmax}_{C_i \in C}[log(P(C_i)) + \sum^t N_{D,t} log P(t|C_i)] \tag{5.3}$$

$P(C_i)$ is a prior over the class space which encodes information about the relative distribution of classes in the training data. The internal representation of each class is a log-scaled version of $P(t|C_i)$, the likelihood of each term in the class, which in turn is given by a smoothed maximum likelihood estimate:

$$P(t|C_i) = \frac{1 + \sum_k^{|D|} N_{D_k,t} P(C_i|D_k)}{n + \sum_i^n \sum_k^{|D|} N_{D_k,t_i} P(C_i|D_k)} \tag{5.4}$$

In all three systems, the representation of a language is a normalized version of the distribution of the sum of the training document vectors for each language. In the case of `TextCat`, the normalization is the rank-order statistics, for `Linguini` it is the vector magnitude, and for `LangDetect` it is the (smoothed) vector sum.

## 5.3 LangID as Supervised Machine Learning

All three systems that we have examined implement a supervised machine learning approach to classification, consistent with modern approaches to LangID (see our literature review in Section 2.2). The key differences between the systems are in the subset of $n$-gram features selected, and the exact choice of learning algorithm. A detailed theoretical comparison between learning algorithms is beyond the scope of this thesis. Instead, we focus our attention on the feature selection methodology, and empirically revisit the issue of learning algorithms in Section 5.5.1.

In supervised machine learning, one of the key assumptions is that labeled training data and unlabeled test data come from the "same distribution". This is sometimes known as the inductive learning hypothesis:

> Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples. (Mitchell 1997:pp.23)

For purposes of this discussion, let us consider a dataset consisting of $n$ instances (documents) $D_1, \cdots, D_n$. Each document $D_i$ in the dataset has an associated label $C^j$ from a label space $C$. For the inductive learning hypothesis to hold, we must assume that the marginal probability $P(C)$ and the conditional probabilities $P(D|C)$ are the

same in the labeled training data and the unlabeled test data. To classify a document $D_i$ under a given model, we calculate the probability $P(D_i|C_j)$ for each label $C_j \in C$, using parameters for $P(C)$ and $P(D|C)$ estimated from the training data, and then classify according to $\text{argmax}_{C_j} P(D_i|C_j)$. Consequently, if $P(C)$ and $P(D|C)$ differ in the training and test data, then we cannot expect our classification of document $D_i$ to be correct. Although we have presented this reasoning from a probabilistic perspective, even in non-probabilistic formulations of supervised machine learning, such as the vector-space model of `Linguini` or the rank-order model of `TextCat`, the same basic intuition holds: the models of a language obtained from the training data should closely approximate the model that would be obtained from the test data if we also had labels for the test data.

In the LangID systems we have examined, the label space $C$ is the set of candidate languages. The three systems can also be interpreted as sharing a common document representation, where each document $D_i$ is a distribution over the space of all possible byte $n$-grams. For tractability, all three systems employ feature selection to trim the space, and although the exact feature subset selected varies according to the system (and the parameters of the feature selection, where applicable), all three systems use some function of term frequency in their filtering of the feature space. In essence, all three systems assume that the conditional term frequencies of the subset of byte $n$-grams they select is the same in the training and the test data. This is a reasonable assumption when the training and test data are a random partitioning of data from a single source, as is the standard practice in evaluation of supervised machine learning. However, if this assumption is violated then we cannot expect the learned classifier

to perform well.

### 5.3.1 Homogeneity in Corpus Linguistics

In statistical terms, *homogeneity* is the attribute of two samples being drawn from the same underlying distribution. As we discussed in the previous section, supervised machine learning assumes homogeneity between training and test data, both in terms of the marginal distribution of features, as well as the distribution conditioned on each class. Homogeneity is an important issue in classical statistics, where it may be of interest to determine if two sets of observations are likely to have come from the same population (e.g. to determine if clinical outcomes are significantly different in a group exposed to a drug under testing when compared to a control group that received a placebo). The classical statistical test of homogeneity is formulated using the $\chi^2$ statistic:

$$\chi^2 = \sum \frac{(O - E)^2}{E} \tag{5.5}$$

where $O$ is an observed value, and $E$ is the expected value. In statistical hypothesis testing, the errors $\frac{O-E}{\sqrt{E}}$ are assumed to be independent and to be identically distributed with a normal distribution, hence the $\chi^2$ statistic is the sum of squares of independent normal random variables. In the $\chi^2$ test of homogeneity, under the null hypothesis it is assumed that the two sets of observations share the same distribution. Hence, the sum of squared differences across all categories is expected to have a $\chi^2$ distribution with $n - 1$ degrees of freedom, where $n$ is the number of categories the data falls into.

Assessing the homogeneity of text is a well-studied problem in corpus-based computational linguistics, and the use of $\chi^2$ for this purpose was studied extensively by Kilgarriff (2001). Kilgarriff (2001) specifically argues that $\chi^2$ hypothesis tests are unsuitable for determining if two corpora are drawn from the same population, because one of the key assumptions is that the count (and hence the error) for each category is independent. In applying a $\chi^2$ test to text, each "category" is a particular word. It is clear that the frequencies of different words in a text cannot be independent of each other; otherwise there could not be any meaning conveyed by the text. It must therefore be the case that the assumption of independence between counts for each category as required for $\chi^2$ hypothesis tests is violated. Kilgarriff described the problem as follows:

> Since words in a text are not random, we know that our corpora are not randomly generated. The only question, then, is whether there is enough evidence to say that they are not, with confidence. (Kilgarriff 2001:pp.236)

The solution proposed by Kilgarriff (2001) is to continue to use the $\chi^2$ statistic, not in the framework of hypothesis testing, but rather simply as a measure of the (dis)similarity between two corpora. The method proposed by Kilgarriff is given as follows:

- divide the corpus into 'slices';
- create two subcorpora by randomly allocating half the slices to each;
- measure the similarity between the subcorpora;
- iterate with different random allocation of slices;

| **Term** | $o_{t,1}$ | $o_{t,2}$ | $e_{t,1}$ | $e_{t,2}$ | **Residual 1** | **Residual 2** |
|---|---|---|---|---|---|---|
| term 1 | 14 | 8 | 13.2 | 8.8 | 0.048 | 0.073 |
| term 2 | 12 | 8 | 12.0 | 8.0 | 0.000 | 0.000 |
| term 3 | 18 | 10 | 16.8 | 11.2 | 0.086 | 0.129 |
| term 4 | 6 | 8 | 8.4 | 5.6 | 0.685 | 1.029 |
| term 5 | 10 | 6 | 9.6 | 6.4 | 0.017 | 0.025 |
| | | | | $\chi^2$ | 2.092 | |
| | | | | CBDF | 0.523 | |

Table 5.3: Example of $\chi^2$ and CBDF calculation for two corpora. $o_{t,x}$ is the count of how often term $t$ occurs in corpus $x$ (i.e. the observed value), and $e_{t,x}$ is the expected value.

- calculate mean and standard deviation over all iterations;

An example of how to calculate $\chi^2$ between two corpora is given in Table 5.3. In this example, we consider two imaginary corpora with only 5 unique terms, that vary in distribution between the two corpora. The expected value for each term is estimated as follows (Kilgarriff 2001:pp.254): If the size of the corpora 1 and 2 are $N_1$ and $N_2$ and term $t$ has observed frequencies $o_{t,1}$ and $o_{t,2}$, then the expected value $e_{t,1} = \frac{N_1 \times (o_{t,1}+o_{t,2})}{N_1+N_2}$ and $e_{t,2} = \frac{N_2 \times (o_{t,1}+o_{t,2})}{N_1+N_2}$. A residual $\frac{(o-e)^2}{e}$ is calculated for each term in each corpus, and the $\chi^2$ statistic is simply the sum of these residuals.

Kilgarriff (2001) applies a normalization by the degrees of freedom, i.e. the number of terms in the vocabulary used for comparison. Kilgarriff (2001) names this metric CBDF, "$\chi^2$(**c**hi-squared) **b**y **d**egrees of **f**reedom". The degrees of freedom for a contingency table of size $m \times n$ is $(m-1)(n-1)$. In our example, $m = 5$ (five terms) and $n = 2$ (two corpora), so the degrees of freedom we use for normalizing is $4 \times 1 = 4$. Intuitively, this metric approximates the average residual per-term, summed over the two corpora. Lower values of the metric imply that on average, the difference be-

tween the term frequencies of terms in the two corpora is lower, and hence the corpora are more similar. Kilgarriff (2001) points out that the absolute values of CBDF are harder to interpret, and suggests that the best practice is to compare CBDF values between different pairs of corpora to determine which pairs are more similar.

### 5.3.2 Assessing Homogeneity of LangID Datasets

In Section 5.3.1 we discussed how Kilgarriff (2001) introduced the notion of homogeneity in order to quantify how similar English-language corpora are to each other by comparing the frequency with which words occur. The problem that we are posing is very similar. Specifically, we wish to quantify how similar corpora of the same language from different sources are to each other. In this section, we adapt Kilgarriff's reasoning and CBDF metric to the problem of assessing the homogeneity of LangID datasets. In the LangID systems we have examined (Chapter 4), documents are not modeled by the frequencies with which words occur, but rather by the frequencies with which particular byte $n$-gram sequences occur (Section 5.1). Like words, $n$-gram sequences are not independent, and the dependence is likely to be greater due to the potential for overlap between sequences. Hence, the same issue in the direct application of $\chi^2$ hypothesis tests exists in applying such a test to determine if the distribution of byte $n$-gram sequences is consistent in different datasets for the same language. Likewise, Kilgarriff's use of CBDF to compare different corpora is directly applicable to the comparison of datasets for LangID, with the modification that rather than computing word frequency lists, we compute byte $n$-gram frequency lists.

Earlier, we noted that in supervised machine learning, both the marginal and

conditional distributions are assumed to be shared between training and test data. We now make use of Kilgarriff's methodology to assess whether this is the case in the LangID datasets we have collected. Differences in marginal distribution are expected, since each dataset contains different quantities of data for different languages (Figure 3.10), and the size of documents also varies between datasets (Table 3.1). Furthermore, such differences can generally be corrected for (e.g. by adjusting the class prior in a Bayesian classifier). Of greater interest is the similarity (or dissimilarity) of the conditional distributions, i.e. the relative frequency of byte $n$-grams in each language across datasets. In order for a language identifier to be applicable across datasets, we would expect that the conditional distribution of each language should be homogeneous across datasets. To test this, we make use of the same subset of data that we used for one-source cross-domain evaluation (Section 4.4.1), where we used the Swedish, Italian, French, Spanish and Danish subset of data across 8 different datasets.

For this experiment, we first determined the top 50,000 byte 4-gram sequences (for simplicity, hereafter we refer to each byte 4-gram as a term) by term frequency across the union of all the data (i.e. 5 languages across 8 datasets), which serves as our initial feature space, approximating the feature space used by the three systems we compared in Chapter 4. Each document was then converted to a 50,000-dimensional vector, each dimension representing the frequency of a particular term. Using this pool of data, we apply the methodology described by Kilgarriff (2001), with the aim of determining the relative homogeneity of documents under this byte 4-gram representation: (1) within the same source of text, and (2) within the same language.

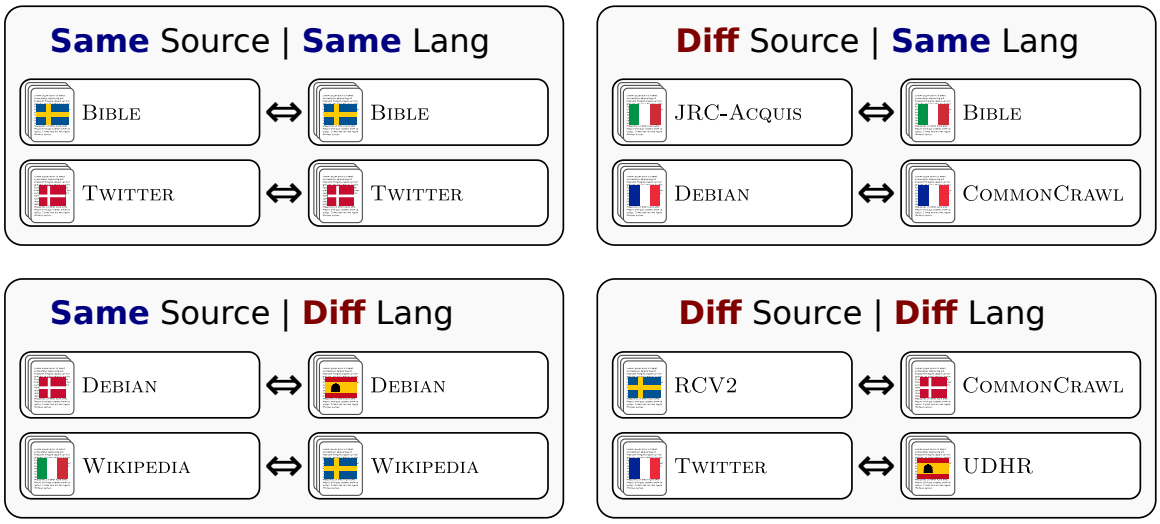



Figure 5.1: Illustration of the division of sub-corpora pairs into 4 distinct bins.

We divide the full data into two halves by randomly assigning each document to either of two partitions (hereafter A and B), stratified by language and dataset, resulting in 80 sub-corpora (8 datasets × 5 languages × 2 partitions). For each sub-corpus we tabulated the marginal term frequency, and then computed $\chi^2$ pairwise between each possible pairing of language-dataset combinations across partitions (i.e. each language-dataset sub-corpus in A is paired with every language-dataset sub-corpus in B in turn, producing 1600 unique pairings). We repeat this procedure 10 times, producing 10 estimates of similarity for each of the 1600 pairings. These 1600 pairings can be broken down into 4 classes (Figure 5.1): (SS), where A and B are sub-corpora from the same text source in the same language (40 pairs), (SD), where the pair differs only in text source (280 pairs), (DS), where the pair differs only in language (160 pairs), and (DD), where the pair differs in both language and text source (1120 pairs). For each of the classes, we calculate the average CBDF across all the pairs in that class, broken down by the language of the first item in the pair. These values are

| | | **Text Source** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Language** | | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* |
| | *same* | 3.4 | 43.6 | 4.4 | 59.3 | 5.3 | 67.1 | 4.7 | 51.9 | 3.5 | 48.5 |
| | *diff* | 58.8 | 85.0 | 61.4 | 96.9 | 64.0 | 98.1 | 58.4 | 89.9 | 58.2 | 88.0 |
| | | da | | es | | fr | | it | | sv | |

Table 5.4: Homogeneity of the distribution of the top 50,000 byte 4-grams by term frequency.

reported in Table 5.4.

For each language in Table 5.4, (SS) is the top-left value, with (SD), (DD) and (DS) following clockwise. As expected, (SS) has the smallest value for all 5 languages (a sub-corpus is most similar to one drawn from the same language and source), and (DD) has the largest value (sub-corpora are most different when both language and source are different). We also observe that (DS) values are closer to (DD) than (SS). This is important as it means that sub-corpora of different languages in the same source are quite different, a pre-requisite for being able to distinguish between languages. The more surprising result is that (SD) values are much closer to (DS) than to (SS). This indicates that for the same language, the variation between source is approximately as large as the variation between languages in the same source. This is evidence that under this subset of byte 4-gram features selected by term frequency, the different datasets are clearly not homogeneous with respect to each language. This has serious implications for the applicability of the inductive learning hypothesis: if the models of the same language that we obtain from different text sources are not homogeneous, then we cannot expect to be able to correctly label documents from a given source on the basis of models of language learned on text from a different source.

### 5.3.3 Transfer Learning

So far, we have shown that the term distributions conditioned on language are not homogeneous with respect to the text source when considering the top 50,000 features by term frequency. This evidence is consistent with the decline in performance we observed in Section 4.4.1 when training a language identifier using data from one source of text and using it to predict the language of documents from a different source of text. In machine learning, the problem of dealing with learning tasks where the marginal probability distribution of the source and target data differs falls under the area of *transfer learning*. We introduced the idea of transfer learning in Chapter 4, when we compared the *in-domain* and *cross-domain* performance of existing LangID systems. So far, we have used the term "domain" interchangeably with "text source". However, in transfer learning terms, Pan and Yang (2010) give a more precise definition of domain:

> A *domain* $D$ consists of two components: a feature space $X$ and a marginal probability distribution $P(X)$, where $X = x_1, \cdots, x_n \in \mathcal{X}$. (Pan and Yang 2010:pp.1346)

Two domains are defined as different if they have "different feature spaces or different marginal probability distributions" (Pan and Yang 2010:pp.1347). As we saw in Section 5.3, under a document representation based on the top 50,000 byte $n$-grams by term frequency, data for the same language from different datasets has a different marginal probability distribution, and should thus be thought of as coming from a different domain. In Section 4.4, we investigated a cross-domain classification task, where we used language-labeled data from different domains to classify documents from a held-out domain. In transfer learning terms, this most closely resembles

*transductive transfer learning*, defined as:

> Given a source domain $D_s$ and a corresponding learning task $T_s$, a target domain $D_t$ and a corresponding learning task $T_t$, *transductive transfer learning* aims to improve the learning of the target predictive function $f_t(\cdot)$ in $D_t$ using knowledge in $D_s$ and $T_s$, where $D_s \neq D_t$ and $T_s = T_t$. In addition, some unlabeled target-domain data must be available at training time. (Pan and Yang 2010:Definition 3)

Pan and Yang (2010) give an excellent overview of the methods proposed in the literature. Most of these methods make use of some statistics of the unlabeled data in the target domain in order to improve the performance on the task. However, this assumption that unlabeled target-domain data is available at training time is not necessarily valid in the case of LangID, and in this thesis we specifically focus on LangID where no additional information is available about the target domain other than the document being classified.[1] This corresponds to a common use case of LangID, where a single document or snippet of text needs to have its language identified, without having additional text from the same domain available.

The majority of transductive transfer learning techniques are not directly applicable when no data is available in the target domain, but still offer insight into how the problem can be tackled. A common theme in transductive transfer learning is the division of the feature space $\mathcal{X}$ into subsets $\mathcal{X}^{specific} \cup \mathcal{X}^{general}$ (Arnold *et al.* 2007), where $\mathcal{X}^{specific}$ is the subspace of features that is strongly associated with a particular domain, whereas $\mathcal{X}^{general}$ is the complementary subspace that is shared amongst the domains. Daumé III and Marcu (2006) present an expectation-maximization al-

[1] Target-domain data is available in the experiments in Section 4.4, but we still classify each instance independently, without making use of any information drawn from the target-domain data collectively.

gorithm that learns three sets of parameters in a maximum-entropy model, one for each of the source and target domains, and a third, "general" set. This approach requires labeled training data in both source and target domains (i.e. inductive transfer learning). Domain-specific features thus have the majority of their weight in one of the domain-specific components, whereas general features have their weight concentrated in the shared, general component. Daumé III (2007) achieves a similar result by transforming the feature set. An augmented input space $\breve{\mathcal{X}}$ is defined, such that given $\mathcal{X} = \mathbb{R}^F$, then $\breve{\mathcal{X}} = \mathbb{R}^{3F}$. Mappings $\Phi^s, \Phi^t : \mathcal{X} \rightarrow \breve{\mathcal{X}}$ are defined by Equation 5.6, where $\mathbf{0} =< 0, 0, \cdots, 0 >\in \mathbb{R}^F$ is the zero vector.

$$\Phi^s(x) =< x, x, \mathbf{0} >, \Phi^t(x) =< x, \mathbf{0}, x > \tag{5.6}$$

Hence, each feature in $\mathcal{X}$ has a copy in $\breve{\mathcal{X}}$ that is shared, as well as a copy that is specific to each domain, resulting in 3 sets of parameters. The expectation is that features that are domain-specific will have most of their weight concentrated in the domain-specific copies, whereas shared features will have most of their weight concentrated in the shared copy. Again, this method is specifically focused on settings where labeled data is available in the target domain.

In contrast to Daumé III and Marcu (2006) and Daumé III (2007), Jiang and Zhai (2006) propose a method for domain adaptation in named entity recognition (NER) that is applicable in situations where no data, labeled or unlabeled, is available in the target domain at training time. Again, the focus is on features that generalize across domains. Rather than consider only a single source domain, Jiang and Zhai (2006) rank features for the same task (NER) in multiple domains. These rankings are then

combined to form an overall ranking of the "generalizability" of the features across multiple domains, which is used as a non-uniform prior to weight a logistic regression model for NER.

None of the methods for domain adaptation we have discussed so far are directly applicable to generalized LangID. The methods of Daumé III and Marcu (2006) and Daumé III (2007) are not suitable because they require labeled data in the target domain. The method of Jiang and Zhai (2006) is unsuitable because named entity recognition is implemented as a binary "in/out" labeling, whereas LangID is a multi-class classification problem. However, we can build on this common theme in transfer learning research, which finds that some features are domain-specific, whereas other features are predictive of the task regardless of the domain. This concept corresponds neatly with some of the motivating ideas given in LangID research to date. For example, Johnson (1993) used language-specific stopword lists, the intuition being that the set of "function words" in a language is independent of where text in the language is drawn from. Grefenstette (1995) examines the use of "short words", arguing that they approximate the set of determiners, conjunctions and prepositions, again with the underlying intuition that these are characteristic of a language regardless of the domain of the document. Giguet (1995) uses "grammatical words", on the premise that such words are characteristic of each language, and differ from one language to the next. The idea of domain-specific features corresponds with confounding factors such as domain-specific markup, which may dominate term-frequency statistics (e.g. in the case of `TextCat`, Section 4.1.1) but have little to no relationship to the actual language of the document. It may also capture more subtle variation such as

differences in topic, where the topics covered make certain terms more prominent in a particular dataset. Our focus is thus to develop a method to identify "general" features for LangID– that is, a feature set that is strongly discriminative between languages regardless of the domain that the data is drawn from. The method must not require unlabeled data in the target domain, as we cannot assume such data is available, and it must be applicable to a multiclass classification problem such as LangID. In the next section, we develop such a method.

## 5.4 Cross-domain Feature Selection

In the previous section, we discussed the supervised machine learning approaches that are used by existing LangID systems, and used the notion of homogeneity from corpus linguistics to argue that the document representation used for LangID violates basic assumptions of the inductive learning hypothesis, which in turn results in poor performance of the systems on the problem of generalized LangID. In this section, we introduce a feature selection approach to building a document representation that is more homogeneous with respect to any given language across multiple domains.

Feature selection is a form of dimensionality reduction, where a document representation in a high-dimensional feature space (e.g. the space of all possible byte 4-grams) is projected into a lower-dimensional feature space. Dimensionality reduction is desirable because it reduces computational requirements, both in terms of run time and memory capacity, particularly for algorithms that are super-linear in the size of the feature space. Furthermore, dimensionality reduction also has another benefit, in that it has the tendency to reduce *overfit* of the training data (Sebastiani 2002).

Sebastiani (2002:pp.15) describes overfit as "the phenomenon by which a classifier is tuned also to the *contingent* characteristics of the training data, rather than just the *constitutive* characteristics of the categories." Our analysis of homogeneity of languages between datasets reveals exactly such a problem with a simple term-frequency based approach to feature selection: the features selected are not only strongly characteristic of specific languages (the *constitutive* characteristic we desire), but also of the specific dataset that they are drawn from (the *contingent* characteristic we wish to eliminate).

So far, the systems we have examined have used feature selection methods that are generally based around term frequency (Chapter 4), and our initial assessment of language homogeneity across datasets was thus implemented using term-frequency feature selection. Feature selection for text classification has a rich literature, and methods that take into account the underlying class information generally perform better than those that do not. In particular, information gain ($\mathbb{IG}$: Quinlan (1986)) has been shown to be particularly suited to feature selection in a multiclass problem setting such as LangID (Yang and Pedersen 1997; Forman 2003).

### 5.4.1 $\mathbb{IG}$: Information Gain

$\mathbb{IG}$ is an information-theoretic measure that was originally developed as a splitting criterion for decision trees (Quinlan 1986). $\mathbb{IG}$ measures the difference in the entropy of the label distribution of a set of instances before and after partitioning the instances by a certain event. When applied to feature selection for text classification tasks, the label set is simply the class space, and the features used are usually the binary

presence/absence of a given term. In this context, $\mathbb{IG}$ quantifies the information obtained for predicting a given class by knowing the presence or absence of a term in a document (Yang and Pedersen 1997). More formally, the information gain of a term $t$ with respect to a class space $C = \{c_i\}_{i=1}^{m}$ is defined as:

$$IG(C,t) = H(C) - H(C|t) \tag{5.7}$$

$$H(C) = -\sum_{i=1}^{m} P(c_i) log P(c_i) \tag{5.8}$$

$$H(C|t) = -P(t)\sum_{i=1}^{m} P(c_i|t) log P(c_i|t) - P(\bar{t})\sum_{i=1}^{m} P(c_i|\bar{t}) log P(c_i|\bar{t}) \tag{5.9}$$

where $t$ and $\bar{t}$ denote the binary presence/absence of a term. $\mathbb{IG}$ is simply the difference between the entropy of the distribution of instances over class labels before and after conditioning on a particular event (Equation 5.7). In our case, the class labels are languages, and the events are the presence ($t$) or absence ($\bar{t}$) of a particular byte $n$-gram sequence. Equation 5.8 uses a standard information-theoretic definition of entropy of a discrete random variable (Cover and Thomas 2006:pp.15), and the entropy after conditioning over a particular feature is given by the weighted average of the entropy of the distribution of languages in documents that contain and do not contain the particular byte $n$-gram sequence (Equation 5.9). Entropy is a measure of the uncertainty inherent in a distribution; a uniform distribution has the highest entropy, and entropy decreases as the outcome becomes more biased. $\mathbb{IG}$ thus quantifies how much a particular event (in our case, the presence/absence of a particular sequence of bytes) tells us about another event (in our case, the language of the document).

| Language | Text Source | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* |
| *same* | 7.2 | 138.7 | 8.4 | 201.4 | 11.8 | 228.8 | 8.6 | 177.3 | 7.5 | 158.6 |
| *diff* | 186.6 | 288.5 | 200.4 | 338.8 | 208.2 | 342.4 | 191.7 | 313.9 | 187.1 | 301.6 |
| | da | | es | | fr | | it | | sv | |

Table 5.5: Homogeneity of the distribution of the top 10,000 byte 4-grams by term frequency.

### 5.4.2 Cross-domain Homogeneity of Language under $\mathbb{IG}$ Feature Selection

In Section 5.3.2, we examined the homogeneity of the 50,000 most frequent byte 4-grams across 5 languages and 8 domains. We found that the variation between texts from the same domain that differed in language was approximately the same as the variation between texts in the same language that were drawn from different domains. In our next set of experiments, we consider the homogeneity of 10,000-feature subsets of this initial 50,000 feature set, in order to gain some insight into the effect of feature selection on the homogeneity of the document representation across languages and domains. In particular, we contrast feature selection based solely on the frequency of individual byte sequences to feature selection that takes into account the information gain of each byte sequence, with respect to the language of the document it is obtained from but also with respect to the domain that the document comes from.

Table 5.5 reports the average homogeneity (as measured by CBDF) for the distribution of the top 10,000 terms by term frequency. We find that the relative results are consistent with those in Table 5.4 (top 50,000 terms by term frequency). Specifically, the average homogeneity for sub-corpora pairs that differ only in domain (SD,

| Language | | Text Source | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* |
| | *same* | 2.9 | 61.5 | 3.8 | 88.7 | 3.4 | 100.2 | 3.5 | 75.1 | 3.0 | 69.7 |
| | *diff* | 165.9 | 184.5 | 187.4 | 221.4 | 188.2 | 220.2 | 179.3 | 198.0 | 170.5 | 193.8 |
| | | da | | es | | fr | | it | | sv | |

Table 5.6: Homogeneity of the distribution of the top 10,000 byte 4-grams by information gain.

top-right) is roughly comparable to the average homogeneity for pairs that differ only in language (DS, bottom-left), and these values fall somewhere between the homogeneity for sub-corpora pairs from the same language and domain (SS, top-left), and the homogeneity for sub-corpora pairs that differ in both language and domain (DD, bottom-right). The absolute values of CBDF have increased throughout, resulting in a larger range of CBDF scores, both in absolute terms as well as in relative terms (i.e. the ratio $\frac{dd}{ss}$ is larger when less features are selected). This seems to indicate that more frequent features also tend to vary more widely in frequency between languages and domains.

Table 5.6 reports the average homogeneity for the distribution of the top 10,000 terms by information gain. In contrast to the results for term frequency (Table 5.5), there is now a distinct difference between the average homogeneity of subcorpora pairs that vary only in domain (SD, top-right), and subcorpora pairs that vary only in language (DS, bottom-left). For the top 10,000 features by $\mathbb{IG}$, the homogeneity for sub-corpora pairs that vary only in language (DS, bottom-left) is now comparable to that for sub-corpora pairs that vary in both language and domain (DD, bottom-right), and these results are consistent across all 5 languages we consider. Although this is an improvement over the term frequency-based selection, we still see that the average

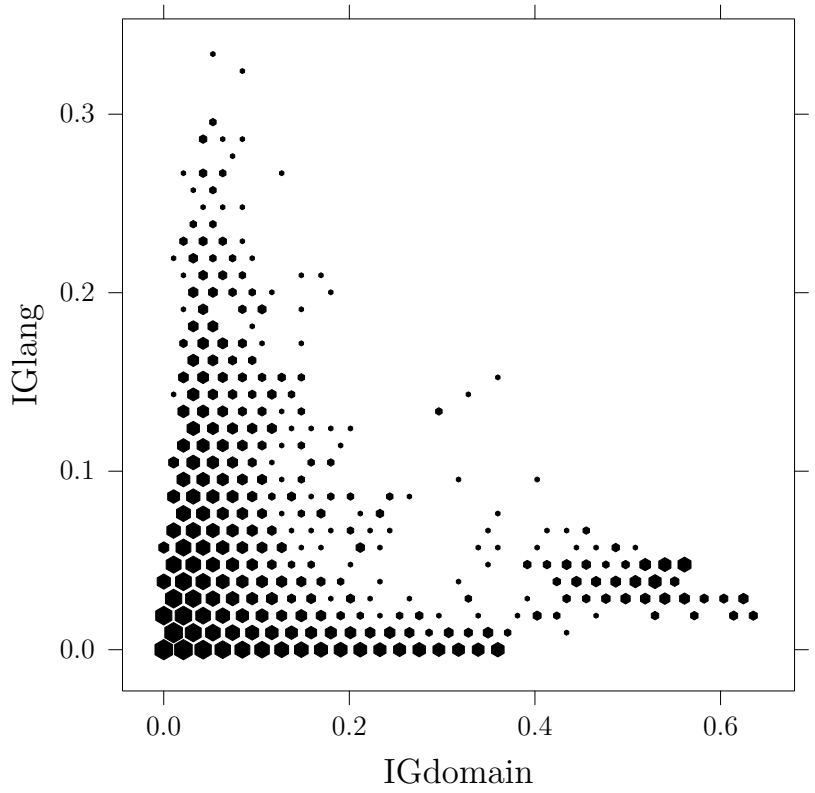



Figure 5.2: Hex-binned scatter plot of $\mathbb{IG}^{lang}$ vs $\mathbb{IG}^{domain}$ for the 50000 most common features by term frequency, in our 5-language subset of data.

homogeneity of sub-corpora of the same language in different domains (SD, top-right) is still markedly higher than that of sub corpora of the same language in the same domain (SS, top-left). This is a problem for generalized LangID, as it means that the distribution of this subset of features still varies substantially by domain, and so a model of a language using data from one domain is unlikely to generalize well to data from another domain.

To understand why languages are not homogeneous across domains under $\mathbb{IG}$-based feature selection, we look towards the relationship between features and the

domain that a document is drawn from. So far, we have used $\mathbb{IG}$ to quantify how informative a given feature is over the set of languages we are considering. We can use a similar approach to quantify how informative a given feature is over the set of domains we draw data from – in other words, how strongly predictive of domain is a given feature? Figure 5.2 shows a scatter plot of the relationship between $\mathbb{IG}$ with respect to language and $\mathbb{IG}$ with respect to domain of each of the 50,000 most common features by term frequency. We observe that features are generally strongly associated with language or domain, but not with both, as evidenced by the absence of data points in the top-right quadrant of the plot. On the basis of our discussion of transfer learning (Section 5.3.3) we are particularly interested in features that are strongly associated with language, but not strongly associated with domain. We observe that in features that are more strongly associated with language (upper half of Figure 5.2), there is some spread in the strength of association with domain. Overall, we would expect that including features that are more strongly associated with particular domain(s) in our document representation will result in a representation that is less homogeneous across domains for any given language. Hence, for our document representation, we want to select features that are strongly associated with language without being strongly associated with domain. As an initial attempt at identifying such features, we introduce the per-feature $\mathbb{IG}_{diff}$ score, defined as:

$$\mathbb{IG}_{diff}(t) = \mathcal{IG}_{lang}(t) - \mathcal{IG}_{domain}(t) \tag{5.10}$$

Table 5.7 reports the average homogeneity for the distribution of the top 10,000 terms by $\mathbb{IG}_{diff}$. We note that the addition of the $\mathbb{IG}_{domain}$ term further improves the average homogeneity of sub-corpora pairs that vary in domain but not in language (SD,

| Language | Text Source | da | | es | | fr | | it | | sv | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* | *same* | *diff* |
| | *same* | 2.4 | 18.1 | 3.0 | 26.4 | 2.5 | 26.7 | 2.6 | 18.8 | 2.4 | 21.1 |
| | *diff* | 132.8 | 116.5 | 157.1 | 138.2 | 151.5 | 132.6 | 145.0 | 126.2 | 139.3 | 124.1 |

Table 5.7: Homogeneity of top 10,000 terms by $\mathbb{IG}_{diff}$.

top-right) with respect to the homogeneity obtained under $\mathbb{IG}_{lang}$ alone, albeit still not being as homogeneous as sub-corpora pairs from the same language and domain (ss, top-left). This suggests that integrating information about both language and domain into the feature selection process should allow us to build a LangID system that is more robust to variation in the domain of the target data than existing systems are, by reducing the extent to which the assumption of homogeneity made in supervised machine learning is violated.

In this section, we introduced information gain ($\mathbb{IG}$), and showed how $\mathbb{IG}$ with respect to both language and domain can be used to select features to produce a document representation that is more homogeneous with respect to language across domains than an equivalent representation based on term-frequency feature selection, and introduced the $\mathbb{IG}_{diff}$ metric for scoring individual features. In the next section, we empirically investigate if a language identifier using $\mathbb{IG}_{diff}$ for feature selection actually outperforms existing LangID systems for cross-domain LangID.

## 5.5 LangID Using Cross-domain Features

In the previous section, we examined the relationship between cross-domain classification and the validity of basic assumptions made in implementing supervised

machine learning. We found that the assumption that training and test data are sampled from the same underlying distribution does not hold when applying standard approaches to LangID across domains, and on the basis of this observation we introduced a document representation that we expect to be robust to cross-domain effects.

Before proceeding to an empirical evaluation of the representation, we first consider some properties of the learning algorithms used by the systems we studied in Chapter 4. To recap, the three systems we considered are: (1) `TextCat`, which is an implementation of the research of Cavnar and Trenkle (1994) and utilizes rank-order statistics, (2) `LangDetect`, which utilizes a multinomial Naive Bayes model, and (3) `Linguini`, which is based on a vector-space model. Hence, the three systems examined represent three broad classes of machine learning algorithms that can be applied to any classification problem. A detailed theoretical comparison of the algorithms is beyond the scope of this thesis, but we will make a short digression to discuss some theoretical aspects as it provides insight into the design of a generalized language identifier.

### 5.5.1 Decision Boundaries

A useful concept to introduce to support our discussion of algorithms is the notion of a "decision boundary". As we discussed in Section 5.1, each document is reduced to a fixed-length vector of numbers, which represent frequency counts over some set of events. These vectors are then used to compute the most likely class for each document. The exact details of the computation vary according to the machine

learning algorithm used, but in all cases we can reason about some properties of the algorithm by considering the *decision boundary* it induces.

Without loss of generality, we can interpret all classification algorithms in the context of a vector-space model. Under such an interpretation, each document is a point in a multi-dimensional feature space. Learning algorithms use sets of training points to generalize certain continuous regions of the feature space as belonging to a particular class; where these regions meet is known as the *decision boundary*. Points on one side of the boundary belong to one class, and points on the other side belong to the other class. The role of the learning algorithm is to select such decision boundaries (also known in machine learning terms as *hypothesis functions* or simply *hypotheses*) that are optimal with respect to some properties of the training data. General-purpose machine learning algorithms such as random forests or SVMs with non-linear kernels are able to fit complex non-linear boundaries to any kind of input data, and are generally thought of as "superior" to "simpler" methods such as a naive Bayes classifier. However, work-to-date in LangID has generally found that the simpler methods perform as well as if not better than the general-purpose methods (see Section 2.2.3). To understand why this might be the case, consider a simple thought experiment: if we take a text document and create a new document by repeating it twice, could the language of the new document ever be different from the language of the old document? If we are willing to assume that the answer is "no", then any boundary where scaling a vector can change the side of the boundary that the vector is on is unsuitable for LangID. General-purpose machine learning algorithms are able to learn such boundaries, and so have the possibility of overfitting training

data in certain ways – for example, it may be the case that in a given dataset, English documents tend to be longer than German documents. Thus, a decision boundary that works well in this dataset may not generalize to a different dataset because it represents the difference between English and German as a function of the length of the document.

In this section, we consider the decision boundaries induced by the three algorithms we are examining in this chapter. Let us begin our analysis by considering the decision boundaries induced by each algorithm in the most trivial case, where we have one training instance for each of two classes, where each instance is a 2-dimensional vector. Our example dataset can be summarized as follows:

$$< x_1, y_1 >: C_1$$

$$< x_2, y_2 >: C_2$$

For `TextCat`, the classification rule is given in Equation 5.1. In this 2-dimensional, 2-class case that we are considering, the decision boundary is given in Equation 5.11.

$$\begin{aligned} &|Rank(x, D) - Rank(x, C_1)| + |Rank(y, D) - Rank(y, C_1)| = \\ &|Rank(x, D) - Rank(x, C_2)| + |Rank(y, D) - Rank(y, C_2)| \end{aligned} \tag{5.11}$$

As there are only two dimensions, there are only two possible permutations of rank over the features. Where both classes have the same ranking (i.e. $Rank(x, C_1) = Rank(x, C_2)$, which in turn implies $Rank(y, C_1) = Rank(y, C_2)$), the decision boundary is not defined. In the remaining cases, the decision boundary lies along $y = x$, since the ranking function is consistent on either side of this boundary – all points on one side of $y = x$ have $y > x$, and all points on the other side have $y < x$. As-

sume that in our example dataset, $x_1 > y_1$ and $x_2 < y_2$. Then, all documents where $x_d > y_d$ will be labeled with $C_1$ and all documents where $x_d < y_d$ will be labeled with $C_2$. In theory, it is possible for $x_d = y_d$, but in our generalized LangID the number of dimensions is typically much larger, and the distribution over features is such that it is extremely improbable for a document to fall exactly on a boundary. This line of reasoning generalizes beyond the 2-dimensional case, and furthermore to any ranklist-based classification, as the relative ranking between any two features can only change when the absolute difference between them changes in sign, i.e the decision boundaries must always occur at the hyperplanes where two features have equal frequency. The difference between ranklist classifiers is thus limited to how each individual "slice" of the hyperspace is labeled.

For the cosine similarity in the vector space model used by `Linguini`, the classification rule is given by Equation 5.2. Training the classifier entails partitioning the vector space into non-overlapping regions, where each region is associated with exactly one language. A new document is thus represented as a point in the vector space, and is assigned the language of the region. The regions are implicitly defined by means of a cosine-based nearest-prototype method. Given training data, all the documents are projected into the vector space, and a centroid for each language is computed as the arithmetic mean of the vectors of all the documents of that language. The decision boundaries in the vector space model thus correspond to a Voronoi decomposition of the vector space, with the decision boundaries between any given pair of centroids given by the hyperplane that is equidistant from both centroids (assuming no other centroids are between them). Equation 5.12 gives the derivation of the

decision boundary for our two-class example.

$$\begin{aligned}
\frac{\sum^t N_{D,t} \cdot N_{C_1,t}}{\sqrt{\sum^t N_{D,t}^2}\sqrt{\sum^t N_{C_1,t}^2}} &= \frac{\sum^t N_{D,t} \cdot N_{C_2,t}}{\sqrt{\sum^t N_{D,t}^2}\sqrt{\sum^t N_{C_2,t}^2}} \\
\frac{x \cdot x_1 + y \cdot y_1}{\sqrt{x^2+y^2}\sqrt{x_1^2+y_1^2}} &= \frac{x \cdot x_2 + y \cdot y_2}{\sqrt{x^2+y^2}\sqrt{x_2^2+y_2^2}} \\
\frac{x \cdot x_1}{\sqrt{x_1^2+y_1^2}} + \frac{y \cdot y_1}{\sqrt{x_1^2+y_1^2}} &= \frac{x \cdot x_2}{\sqrt{x_2^2+y_2^2}} + \frac{y \cdot y_2}{\sqrt{x_2^2+y_2^2}} \\
\frac{x \cdot x_1}{\sqrt{x_1^2+y_1^2}} - \frac{x \cdot x_2}{\sqrt{x_2^2+y_2^2}} &= \frac{y \cdot y_2}{\sqrt{x_2^2+y_2^2}} - \frac{y \cdot y_1}{\sqrt{x_1^2+y_1^2}} \\
x \cdot (\frac{x_1}{\sqrt{x_1^2+y_1^2}} - \frac{x_2}{\sqrt{x_2^2+y_2^2}}) &= y \cdot (\frac{y_2}{\sqrt{x_2^2+y_2^2}} - \frac{y_1}{\sqrt{x_1^2+y_1^2}}) \\
y &= \frac{\frac{x_1}{\sqrt{x_1^2+y_1^2}} - \frac{x_2}{\sqrt{x_2^2+y_2^2}}}{\frac{y_2}{\sqrt{x_2^2+y_2^2}} - \frac{y_1}{\sqrt{x_1^2+y_1^2}}} \cdot x
\end{aligned} \tag{5.12}$$

Finally, the classification rule for the multinomial naive Bayes model used by `LangDetect` is given in Equation 5.3. In our simplified two-instance, two-class, two-dimensional example, $P(C_1) = P(C_2)$, so the decision boundary between the two classes is given in Equation 5.13.

$$\begin{aligned}
\sum^t N_{D,t} log P(t|C_1) &= \sum^t N_{D,t} log P(t|C_2) \\
x \cdot log P(x|C_1) + y \cdot log P(y|C_1) &= x \cdot log P(x|C_2) + y \cdot log P(y|C_2) \\
y &= \frac{log P(x|C_1) - log P(x|C_2)}{log P(y|C_2) - log P(y|C_1)} \cdot x \\
y &= \frac{log \frac{P(x|C_1)}{P(x|C_2)}}{log \frac{P(y|C_2)}{P(y|C_1)}} \cdot x \\
y &= \frac{log \frac{(1+x_1)(2+x_2+y_2)}{(1+x_2)(2+x_1+y_1)}}{log \frac{(1+y_2)(2+x_1+y_1)}{(1+y_1)(2+x_2+y_2)}} \cdot x
\end{aligned} \tag{5.13}$$

From this analysis, we notice two key properties shared by the decision boundaries of all three algorithms. Firstly, the decision boundary is linear (it can be expressed

as $y = m{\cdot}x + c$), and secondly, the intercept term $c$ is 0. Together, these two properties give decision boundaries that are independent of the magnitude of the vector, i.e. the classifier learned is unaffected by the absolute length of the document, but is only influenced by the relative distribution of terms therein. This is the characteristic of decision boundaries suitable for LangID that we identified through our thought experiment at the beginning of this section. The relatively simple decision boundaries learned by the most successful algorithms in LangID match the expected characteristics of the decision boundaries for LangID.

This analysis also reveals an interesting property of the ranklist-based method of `TextCat`: the range of possible decision boundaries is much less expressive than that learned by the algorithms underlying `Linguini` and `LangDetect`. Both the vector-space model and the multinomial naive Bayes model can learn the same boundary as the ranklist-based model of `TextCat`, but only under very specific circumstances – i.e. where the gradient $m$ of the decision boundary is 1. It is not possible to know if this is an advantage or a disadvantage, because this depends on the nature of the problem itself; if the "true" decision boundaries coincide with the limited set that `TextCat` can express, then we should expect that `TextCat` would produce a better fit than the other algorithms with less training data, which would be expected to overfit. However, if the "true" boundaries are distant from those that `TextCat` can express, `TextCat` will underfit the training data and the other algorithms would be expected to do better. There does not appear to be any obvious reason why either should be the case for generalized LangID.

Table 5.8 provides a side-by-side comparison of the decision boundaries of the

$$y = \frac{\frac{x_1}{\sqrt{x_1^2+y_1^2}} - \frac{x_2}{\sqrt{x_2^2+y_2^2}}}{\frac{y_2}{\sqrt{x_2^2+y_2^2}} - \frac{y_1}{\sqrt{x_1^2+y_1^2}}} \cdot x \qquad\qquad y = \frac{log\frac{(\alpha+x_1)(2\cdot\alpha+x_2+y_2)}{(\alpha+x_2)(2\cdot\alpha+x_1+y_1)}}{log\frac{(\alpha+y_2)(2\cdot\alpha+x_1+y_1)}{(\alpha+y_1)(2\cdot\alpha+x_2+y_2)}} \cdot x$$

Table 5.8: Decision boundaries for VSM classifier (left) and NBM classifier (right), in the two-class case where $C_1 :< x_1, y_1 >$ and $C_2 :< x_2, y_2 >$. $\alpha$ is a smoothing parameter for NBM.

vector space model used by `Linguini` and the naive Bayes model used by `LangDetect` respectively. The naive Bayes classifier includes a smoothing parameter $\alpha$, which functions as a uniform Bayesian prior for the estimate of the relative frequency of any given feature in the feature set. $\alpha$ is commonly set to 1 to provide what is known as Laplacian smoothing, and greater values of $\alpha$ push the model towards a uniform distribution over features. When inspected side-by-side, the two decision boundaries appear to be remarkably similar, particularly when we note that in the decision boundary for NBM includes a division in a logarithm, which can be re-written as a difference between logarithms.

Figure 5.3 illustrates the relationship between the gradients of the naive Bayes model used by `LangDetect` and the vector space model used by `Linguini`. For each data point in Figure 5.3, two random points are sampled in a 2-dimensional vector space, and each point is treated as the sole instance of a particular class. In practice, this has the same effect as sampling a number of points for each class and then computing the centroids thereof. The X-axis value of each data point is the gradient of the decision boundary between the two classes computed by the vector space model (Equation 5.12), and the Y-axis value is the gradient of the decision boundary

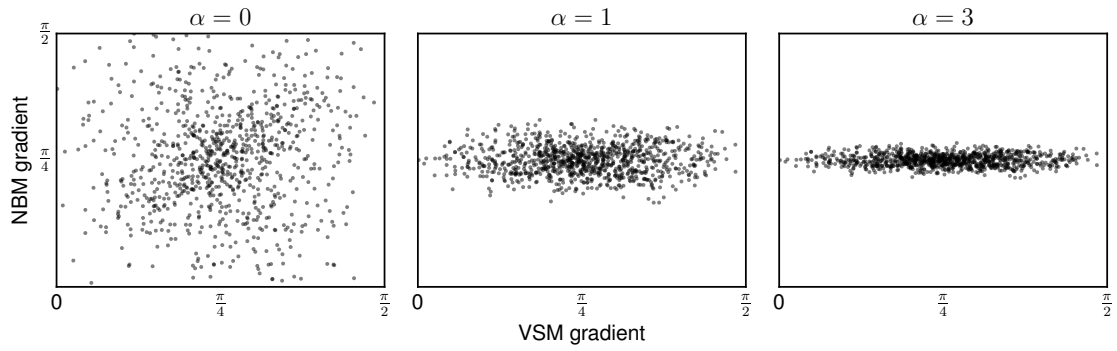



Figure 5.3: Relationship between gradient of VSM classifier and NBM classifier for randomly sampled point pairs in a 2-d vector space. Axes are scaled in radians.

computed by the naive Bayes classifier (Equation 5.13). Each panel corresponds to a different value of the smoothing parameter for the naive Bayes model. We observe that without smoothing, the two decision boundaries appear to be completely uncorrelated, which implies that the differences between them depend entirely on the underlying dataset. Smoothing compressed the range of the gradient of the naive Bayes classifier, which tends towards $\frac{\pi}{4}$ as the degree of smoothing increases. However, this does not affect the correlation between gradients of the naive Bayes classifier and the vector space model, which remains very close to zero regardless of the smoothing factor. We note that a gradient of $\frac{\pi}{4}$ corresponds to a decision boundary of $y = x$, which is the decision boundary for the ranklist model of `TextCat`.

In summary, the decision boundaries of the three learning algorithms we consider all have in common that they are independent of the length of the document, a property that we have argued is desirable for purposes of LangID. However, the three algorithms differ substantially in terms of the types of boundaries they can learn. The ranklist-based algorithm of `TextCat` is substantially less expressive than the vector-

space model of `Linguini` or the naive Bayes model of `LangDetect`. Furthermore, the decision boundaries learned by the vector-space model and the naive Bayes model appear completely uncorrelated. From a theoretical perspective, there is no clear reason to favor one algorithm over another, so we now proceed to an empirical evaluation of the use of each algorithm in combination with the document representations based on information gain that we developed in Section 5.4, to see if any algorithm is particularly suited to generalized LangID using such a representation.

### 5.5.2 Empirical Comparison of Algorithms

In Section 5.5.1, we discussed the theoretical relationship between the different learning algorithms used by the systems we compared in Chapter 4. We showed how all three algorithms were insensitive to the length of a document in making a classification, how the ranklist-based algorithms are less expressive, and how the naive Bayes model and vector space models can learn very different decision boundaries for the same data. However, we concluded that on the basis of our theoretical analysis there was no reason to favor any particular algorithm for the purposes of LangID. In order to determine which algorithm is most suited to generalized LangID, we now proceed to an empirical evaluation of the three algorithms on cross-domain LangID. We use the same evaluation setup that we used in Section 4.4.2. Like in Section 4.4.2, we use the whole of Twitter-trn and UDHR-trn, but downsample each of JRC-Acquis, Bible, CommonCrawl, Debian, RCV2, SETimes and Wikipedia to 10% of the original data, randomized and stratified by language.

We investigate each of the three algorithms in combination with the three meth-

| | `TextCat` | $\mathbb{TF}$ | $\mathbb{IG}_{lang}$ | $\mathbb{IG}_{diff}$ |
|---|---|---|---|---|
| JRC-Acquis | 0.618 | 0.254 | 0.725 | 0.990 |
| Bible | 0.607 | 0.493 | 0.500 | 0.744 |
| CommonCrawl | 0.191 | 0.420 | 0.690 | 0.796 |
| Debian | 0.641 | 0.371 | 0.593 | 0.658 |
| RCV2 | 0.368 | 0.639 | 0.791 | 0.801 |
| SETimes | 0.639 | 0.545 | 0.681 | 0.775 |
| Twitter | 0.563 | 0.330 | 0.479 | 0.613 |
| UDHR | 0.538 | 0.382 | 0.546 | 0.634 |
| Wikipedia | 0.463 | 0.313 | 0.437 | 0.475 |
| average | 0.514 | 0.416 | 0.605 | 0.721 |

Table 5.9: Macro-averaged F-score for different feature sets using RanklistModel.

ods for feature selection that we have considered so far. To recap, the algorithms are the vector-space classifier used by `Linguini` (hereafter VectorSpaceModel), the multinomial naive Bayes classifier used by `LangDetect` (hereafter LikelihoodModel), and the nonparametric term ranking model used by `TextCat` (hereafter RanklistModel). The three feature selection methods we have considered are term frequency ($\mathbb{TF}$), per-language information gain ($\mathbb{IG}_{lang}$), and information gain difference between language and data source ($\mathbb{IG}_{diff}$). We use the exact same feature set from the experiments on homogeneity in Section 5.3.2. Specifically, we use each feature selection method to select the top 10,000 byte 4-grams out of the top 50,000 byte 4-grams by term frequency. For each combination of feature set and algorithm, we train a model for each data source. As in Section 4.4.2, this is structured like a cross-validation experiment, where each "partition" is a data source, so the "train" data for a target data source is the union of the data from all sources excluding the target.

Table 5.9, Table 5.10 and Table 5.11 report the macro-averaged F-score for Ranklist-

| | `Linguini` | $\mathbb{TF}$ | $\mathbb{IG}_{lang}$ | $\mathbb{IG}_{diff}$ |
|---|---|---|---|---|
| JRC-Acquis | 0.801 | 0.071 | 0.716 | 0.975 |
| Bible | 0.607 | 0.361 | 0.484 | 0.744 |
| CommonCrawl | 0.680 | 0.401 | 0.484 | 0.761 |
| Debian | 0.530 | 0.258 | 0.465 | 0.644 |
| RCV2 | 0.421 | 0.233 | 0.475 | 0.825 |
| SETimes | 0.615 | 0.303 | 0.401 | 0.729 |
| Twitter | 0.399 | 0.232 | 0.416 | 0.542 |
| UDHR | 0.487 | 0.104 | 0.243 | 0.664 |
| Wikipedia | 0.283 | 0.244 | 0.335 | 0.395 |
| average | 0.536 | 0.245 | 0.447 | 0.698 |

Table 5.10: Macro-averaged F-score for different feature sets using VectorSpaceModel.

| | `LangDetect` | $\mathbb{TF}$ | $\mathbb{IG}_{lang}$ | $\mathbb{IG}_{diff}$ |
|---|---|---|---|---|
| JRC-Acquis | 0.914 | 0.710 | 0.954 | 0.991 |
| Bible | 0.766 | 0.548 | 0.641 | 0.791 |
| CommonCrawl | 0.224 | 0.298 | 0.355 | 0.577 |
| Debian | 0.748 | 0.484 | 0.668 | 0.675 |
| RCV2 | 0.793 | 0.611 | 0.750 | 0.781 |
| SETimes | 0.713 | 0.528 | 0.719 | 0.785 |
| Twitter | 0.638 | 0.424 | 0.633 | 0.690 |
| UDHR | 0.655 | 0.324 | 0.483 | 0.752 |
| Wikipedia | 0.460 | 0.328 | 0.395 | 0.446 |
| average | 0.657 | 0.473 | 0.622 | 0.721 |

Table 5.11: Macro-averaged F-score for different feature sets using LikelihoodModel.

Model, VectorSpaceModel and LikelihoodModel, respectively. For each system, we also reproduce the cross-domain results from Section 4.4.2 for comparison. For all three learning algorithms, we can see that the $\mathbb{IG}_{lang}$ feature set outperforms $\mathbb{TF}$, and that $\mathbb{IG}_{diff}$ in turn outperforms $\mathbb{IG}_{lang}$. These results are consistent with our expectations from our earlier analysis of homogeneity (Section 5.3.1), where we

analyzed the homogeneity of languages with respect to different sources of text. The feature selection method that produces the most homogeneous feature set ($\mathbb{IG}_{diff}$) consistently outperforms the less homogeneous feature sets across different learning algorithms and different target domains. We also note that, averaged across the set of 9 target domains, the $\mathbb{IG}_{diff}$ feature set outperforms the off-the-shelf system retrained using the same training data. This empirically demonstrates the importance of integrating "domain" information in the feature selection process for building a language identifier that is robust to variation in language across different sources of text.

In terms of choice of learning algorithm, our results suggest that RanklistModel and LikelihoodModel are better suited for generalized LangID than VectorSpaceModel. However, the choice between RanklistModel and LikelihoodModel is harder, as the average result across all datasets is very close. Examining results on a per-dataset level, LikelihoodModel is better than RanklistModel on most datasets, the exceptions being CommonCrawl and RCV2. Notably, these are datasets where `TextCat` performed most poorly when re-trained on in-domain data (Section 4.3). The issue here is related to the issue that we observed in Table 4.2: the most frequent $n$-grams in these datasets are due to the markup present in the particular source of text, and are thus not strongly predictive of language. `TextCat` select features on a per-language basis, independently for each language. In CommonCrawl and RCV2, this results in the same high-frequency "noise" being selected for each language, illustrated in Table 4.2. The feature selection methods that we have examined so far in this chapter have all been global, meaning a single set of features is selected across all languages. In this particular instance, despite

the highest-ranked features by $\mathbb{TF}$ being "noise", enough features have been selected that some language-specific features have been retained. This explains the big gap in accuracy between `TextCat` and RanklistModel with $\mathbb{TF}$ feature selection. Thereafter, when we actually take language information into account, less "noise" features are selected and as such the accuracy further increases.

### 5.5.3 Global vs Local Feature Selection

Figure 5.4 shows a per-dataset boxplot of the distribution of F-scores over the languages in each dataset, broken down by classifier. The results for using $\mathbb{IG}_{diff}$ feature selection with each of the learning algorithms are presented in the `IG`$x$ columns. One issue that we observe is that there is a great deal of variance in the per-language F-score of each system. So far, the methods of feature selection that we have examined ($\mathbb{TF}$, $\mathbb{IG}_{lang}$, $\mathbb{IG}_{diff}$) have all been *global* methods, in that a score is calculated per-feature and the top scoring features are retained. Figure 5.4 shows that there is a substantial variance in the F-score attained per-language, with some languages achieving near-perfect results, whereas others fare much more poorly. To address this disparity, we now investigate the use of *local* feature selection. In contrast to *global* feature selection, in *local* feature selection a score is computed for each feature *per-class*, and a number of features are selected for each class to be added to the final feature set. The method described by Cavnar and Trenkle (1994) is effectively a local feature selection, as the top features by frequency are selected on a per-language basis. Our best-performing metric so far has been $\mathbb{IG}_{diff}$, which we defined as the difference between information gain with respect to language and information gain

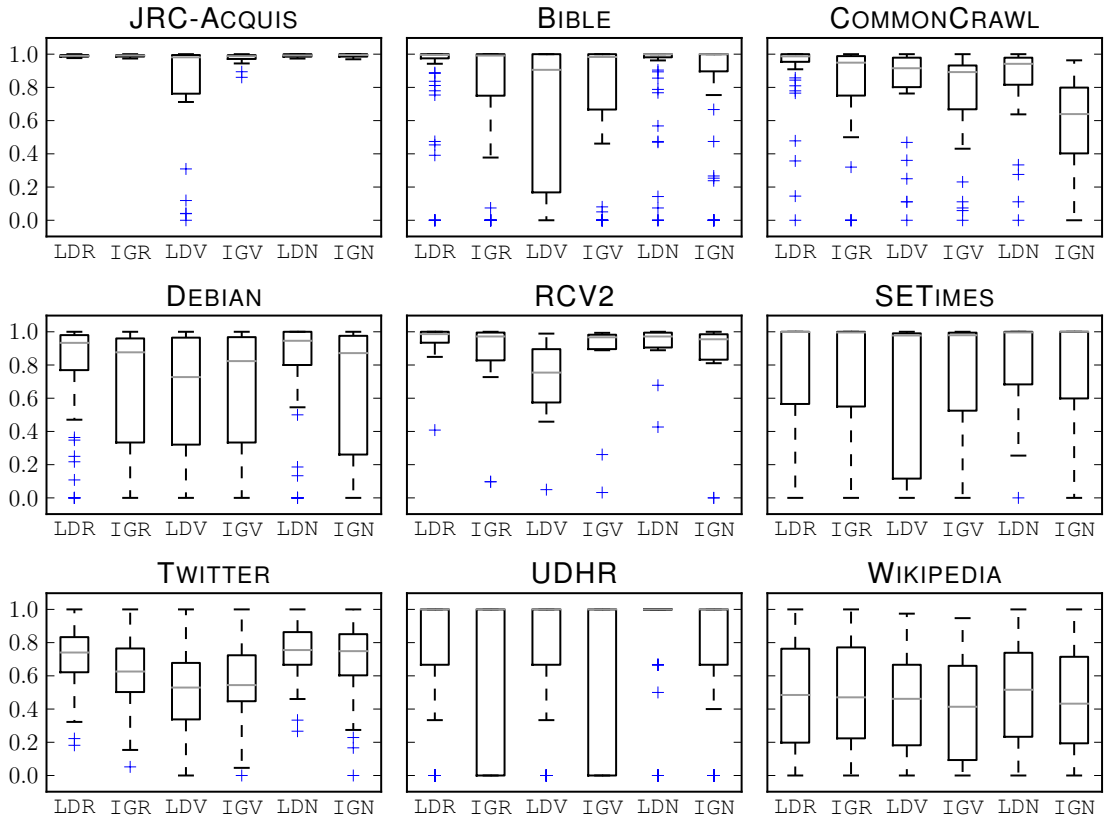



Figure 5.4: Per-dataset boxplot of distribution of F-scores over the languages in each dataset, broken down by classifier. Each column corresponds to one classifier: `LDR`=RANKLISTMODEL using $\mathbb{LD}$ features, `LDV`=VECTORSPACEMODEL using $\mathbb{LD}$ features, `LDN`=LIKELIHOODMODEL using $\mathbb{LD}$ features, `IGR`=RANKLISTMODEL using $\mathbb{IG}_{diff}$ features, `IGV`=VECTORSPACEMODEL using $\mathbb{IG}_{diff}$ features, and `IGN`=LIKELIHOODMODEL using $\mathbb{IG}_{diff}$ features.

with respect to data source on a per-feature basis (Equation 5.10). Since features are scored independently of each other, a global feature selection such as $\mathbb{IG}_{diff}$ risks selecting features associated with languages that are more distinct, as such languages would have features with inherently higher information gain. For example, if a par-

ticular set of features occurs in only one language (e.g. particular characters, such as the Korean Hangul script), these features would all have a higher information gain with respect to language than features such as sequences of letters in Cyrillic script that may be characteristic of several languages. As a result, a global feature selection risks selecting a disproportionate number of features that are very strongly associated with a single language. This explanation is supported by our results in Figure 5.4, where the systems using global feature selection (in this case $\mathbb{IG}_{diff}$ in the `IG`$x$ columns) perform very well only on relatively few languages. To mitigate this, we introduce a *local* variant of $\mathbb{IG}_{diff}$. We continue to use the information gain with respect to the data source as a discounting factor, but instead of using information gain with respect to the set of all languages, we score each feature using information gain binarized with respect to each language in turn. We thus define the $\mathbb{LD}$ score as:

$$\mathbb{LD}(t, l) = \mathcal{IG}_{lang}(t, l) - \mathcal{IG}_{domain}(t)$$

Whereas for $\mathbb{TF}$, $\mathbb{IG}_{lang}$ and $\mathbb{IG}_{diff}$ we selected a fixed number of features overall, for $\mathbb{LD}$ we select a number of features per-language. Cavnar and Trenkle (1994) use language profiles consisting of the top 400 features by term frequency per-language. However, the top-ranked features by term frequency are not necessarily informative of language and thus a smaller number of features may be needed to attain the same accuracy when using a feature selection that is sensitive to the target labels. We verify this by examining the effect of the number of features selected per-language on the accuracy of the classifier under each of the learning algorithms. We experiment with selecting 50 to 300 features per-language in 50-feature increments. The results

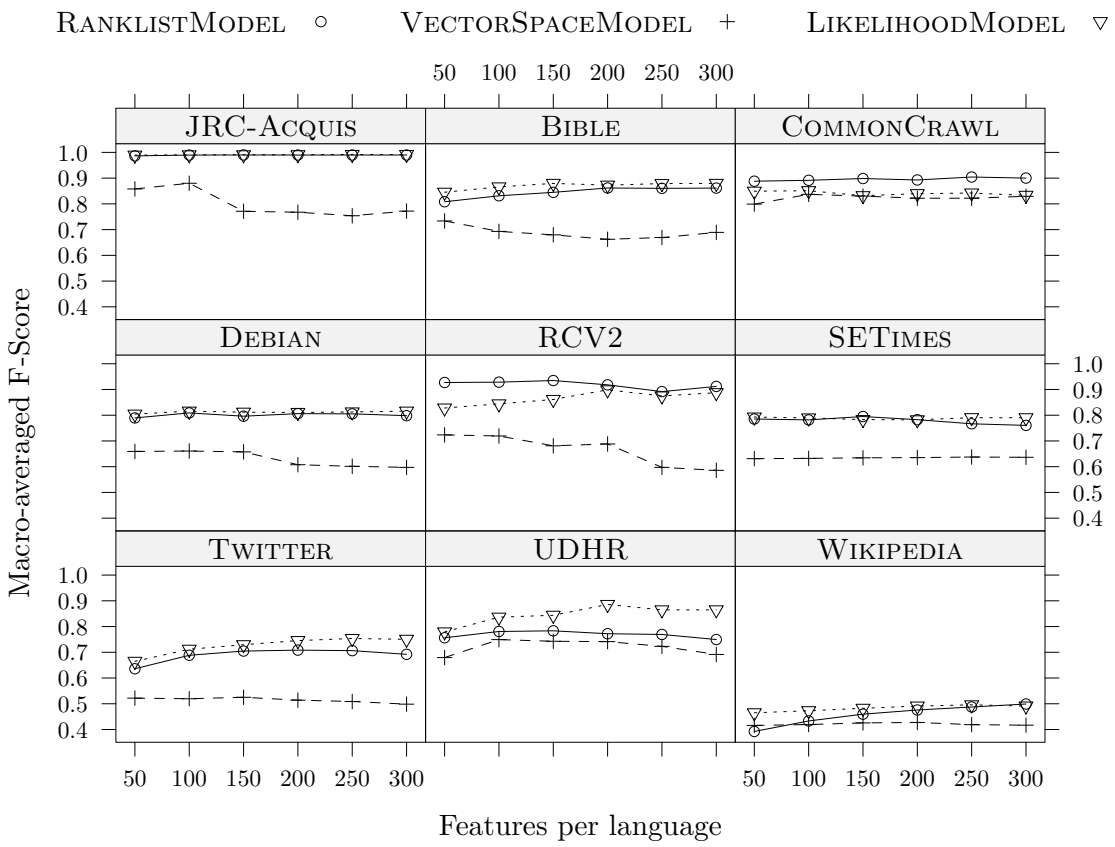



Figure 5.5: Effect of number of features selected per-language in $\mathbb{LD}$ feature selection.

of this experiment are presented in Figure 5.5. We immediately note that VectorSpaceModel performs consistently worse than RanklistModel and LikelihoodModel; we had also noted this under the other feature selection methods we examined. There is much less difference between RanklistModel and LikelihoodModel; RanklistModel again performs better on CommonCrawl and RCV2, but LikelihoodModel performs better in Bible, Twitter and UDHR. On the remaining datasets, the results are effectively indistinguishable. Another trend that we notice is that increasing the number of features per language has a very lim-

ited effect on accuracy; at 50 features per language, the accuracy of RanklistModel and LikelihoodModel on most datasets has already reached a maximum. The only datasets where a substantial increase in accuracy is observed with increased feature count per-language are Twitter and UDHR. This is likely due to the extreme nature of the two datasets; Twitter has the shortest documents, and UDHR has a relatively high number of languages. Short documents result in sparse feature vectors, and increase the possibility of documents containing no features in the feature set. Increasing the number of features in the feature space thus helps to ensure coverage in the feature set. Having a high number of languages increases the possibility of similar languages being included, and increasing the number of features increases the likelihood of being able to distinguish between the similar languages. Another dataset with a high number of languages is Wikipedia, and here we also observe an increase in accuracy with increasing number of features per language, though the increase is not as great.

Figure 5.4 also includes results for $\mathbb{LD}$ feature selection in the `LD`$x$ columns. For RanklistModel and LikelihoodModel, the transition from global to local feature selection has had the desired effect: the overall variance in per-language F-score has been reduced for all datasets, and this effect is particularly prominent in CommonCrawl, Debian and RCV2. Overall, this also leads to better macro-averaged F-scores when using $\mathbb{LD}$ feature selection rather than $\mathbb{IG}_{diff}$.

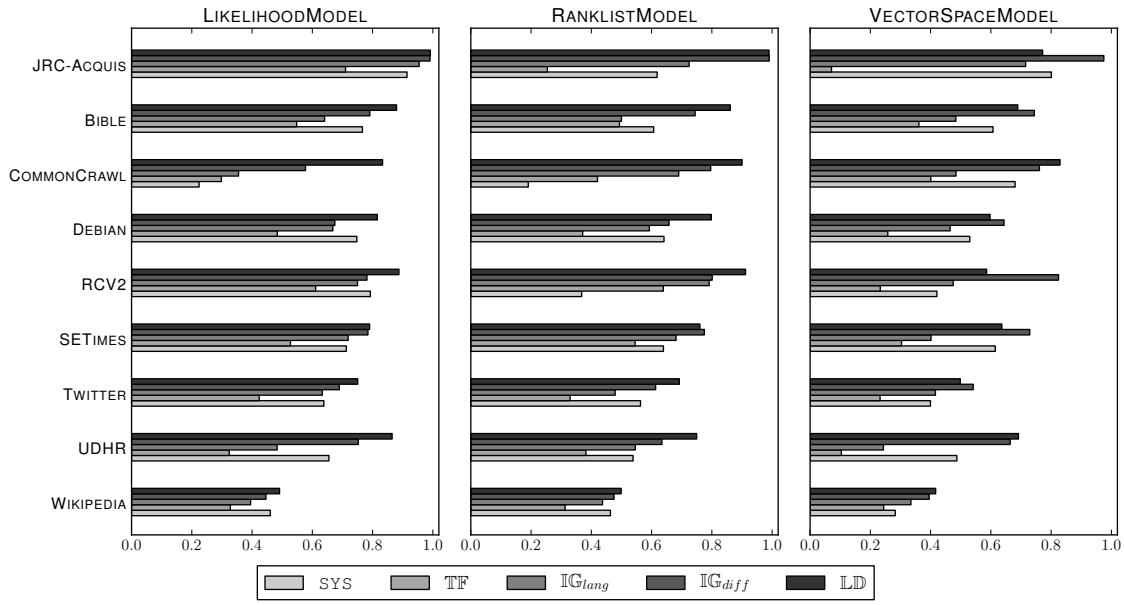



Figure 5.6: Summary of results for cross-domain training of language identifiers using different algorithms and different feature selection methods. `SYS` in each column is the result for the off-the-shelf system using the given algorithm (`LangDetect` for LIKELIHOODMODEL, `TextCat` for RANKLISTMODEL, `Linguini` for VECTORSPACEMODEL).

### 5.5.4 Summary of Results by Feature Selection Method

In this section, we have investigated methods for feature selection that take into account both the language of the training data as well as the source of text. The results of our investigation are summarized in Figure 5.6, which displays the macro-averaged F-score for each feature selection method in combination with each learning algorithm. All the results in Figure 5.6 make use of the exact same training and test data. The `SYS` bars are the results obtained by re-training the off-the-shelf systems with the union of the other out-of-domain datasets, and are replicated from

Section 4.4.2. Overall, the results are clear: there is a distinct improvement as we progress between the different feature selection methods that we discussed, and the best results are obtained by a feature selection method ($\mathbb{LD}$) that selects features *locally* (i.e. per-language), and takes into account both the source of the text and the language. Of the three learning algorithms, VectorSpaceModel generally performs worse than LikelihoodModel and RanklistModel using all the feature selection methods that we tested, and furthermore it sometimes performs better with the global method $\mathbb{IG}_{diff}$ than the local method $\mathbb{LD}$. Between LikelihoodModel and RanklistModel the difference in accuracy is minimal, with LikelihoodModel slightly outperforming RanklistModel. However, the classifiers based on the feature selection methods we have developed consistently outperform the standard language identifiers we have examined when re-trained on the same data.

### 5.5.5 Byte 4-grams vs $n$-grams

So far, we have focused on subsets of the space of all possible byte 4-grams, because it facilitated the analysis and discussion of homogeneity (Section 5.3.1). However, in LangID research to date, it is also common to use a mixed-order $n$-gram representation. A classic example of this is Cavnar and Trenkle (1994), which uses a mixed $n$-gram representation for $1 \leq n \leq 4$. This means that all sequences of 1 to 4 bytes are considered as candidates in the feature selection stage. We carried out an experiment to compare the effectiveness of byte 4-grams to byte $n$-grams in a cross-domain setting using $\mathbb{LD}$ feature selection. As previously, we considered the three learning

| | LikelihoodModel | | RanklistModel | | VectorSpaceModel | |
|---|---|---|---|---|---|---|
| | 4-gram | $n$-gram | 4-gram | $n$-gram | 4-gram | $n$-gram |
| JRC-Acquis | 0.991 | 0.988 | 0.990 | 0.990 | 0.768 | 0.480 |
| Bible | 0.873 | 0.883 | 0.862 | 0.832 | 0.662 | 0.687 |
| CommonCrawl | 0.840 | 0.814 | 0.893 | 0.935 | 0.822 | 0.195 |
| Debian | 0.811 | 0.799 | 0.806 | 0.748 | 0.608 | 0.605 |
| RCV2 | 0.899 | 0.925 | 0.918 | 0.919 | 0.688 | 0.664 |
| SETimes | 0.782 | 0.753 | 0.783 | 0.786 | 0.635 | 0.638 |
| Twitter | 0.745 | 0.769 | 0.708 | 0.731 | 0.514 | 0.420 |
| UDHR | 0.886 | 0.812 | 0.772 | 0.745 | 0.741 | 0.665 |
| Wikipedia | 0.491 | 0.505 | 0.476 | 0.455 | 0.427 | 0.420 |
| average | 0.813 | 0.805 | 0.801 | 0.793 | 0.652 | 0.531 |

Table 5.12: Comparison of macro-averaged F-score for each learning algorithm using $\mathbb{LD}$ to select 200 byte 4-grams per language (4-gram) and 200 byte $n$-grams per-language for $1 \leq n \leq 4$ ($n$-gram).

algorithms LikelihoodModel, RanklistModel and VectorSpaceModel, using $\mathbb{LD}$ feature selection and selecting between 50 and 300 features per language in 50-feature increments. Similarly to byte 4-grams, we found that the difference between selecting different numbers of features per language in the range that we have examined is minimal. Table 5.12 shows a comparison between selecting 200 byte 4-grams and byte $n$-grams per-language using $\mathbb{LD}$. Overall, there appears to be an advantage in using a byte 4-gram representation rather than a mixed $n$-gram representation. The actual difference is fairly small in the case of LikelihoodModel and RanklistModel, and more pronounced for VectorSpaceModel. There is some variation in performance between text sources, where for some sources the byte $n$-gram representation is preferable, whereas for other sources the byte 4-gram representation results in a more accurate classifier. We explore reasons for this in more detail in our error analysis in Section 5.6.3.

| | **English** | **French** | **Italian** | **German** | **Dutch** | **Japanese** |
|---|---|---|---|---|---|---|
| character | ␣its | fair | ␣ess | llte | ijze | □□□ |
| byte | 20 69 74 73 | 66 61 69 72 | 20 65 73 73 | 6C 6C 74 65 | 69 6A 7A 65 | AE 9F E8 A1 |
| character | ␣hav | du␣c | colt | esch | n␣zi | □– |
| byte | 20 68 61 76 | 64 75 20 63 | 63 6F 6C 74 | 65 73 63 68 | 6E 20 7A 69 | 92 E2 88 92 |
| character | ␣oth | ␣vou | iato | habe | emee | ␣ フ |
| byte | 20 6F 74 68 | 20 76 6F 75 | 69 61 74 6F | 68 61 62 65 | 65 6D 65 65 | 20 E3 83 95 |
| character | ed␣a | ␣jou | nzio | wurd | ehee | –□ |
| byte | 65 64 20 61 | 20 6A 6F 75 | 6E 7A 69 6F | 77 75 72 64 | 65 68 65 65 | E2 88 92 EF |
| character | out␣ | r␣à | senz | ␣wer | euwe | □ 5 |
| byte | 6F 75 74 20 | 72 20 C3 A0 | 73 65 6E 7A | 20 77 65 72 | 65 75 77 65 | 8E EF BC 95 |

Figure 5.7: Language-indicative 4-byte sequences selected by $\mathbb{LD}$. □ is used as a placeholder for byte sequences that are incomplete codepoints and thus have no corresponding glyph.

### 5.5.6 Language-indicative Byte Sequences

We have developed a document representation based on feature selection that outperforms the native document representation of `TextCat`, `LangDetect` and `Linguini` when trained on data that is from multiple different text sources. Our feature selection technique is based on integrating not just information about the language(s) that features occur in, but also about the source of text that the training documents are drawn from. Figure 5.7 gives examples of the top-5 features per-language as scored by $\mathbb{LD}$ over a subset of 6 languages. These features can be thought of as *language-indicative byte sequences*, as their presence in a text strongly suggests that the document is written in a specific language. Examining the features, we see that

they provide empirical evidence to support previous work in LangID that has used externally-specified word fragments as being indicative of languages, such as the characteristic word tables of Ingle (1976), the word fragments of Dunning (1994), or the grammatical words of Giguet (1995). In contrast to this previous work, our method has discovered these features entirely through a data-driven process, with no manual input beyond the language that an entire document is written in. This means that the method can be applied to languages that have limited linguistic resources such as function word lists. We also find that the best features represent a mixture of ideas from previous work, with prefixes and suffixes but also sequences that occur in the middle of words.

Another interesting phenomenon is the inclusion of sequences which span multiple words, evidenced by the presence of a space. In this section, we focused primarily on 4-byte sequences, which may artificially constrain the prefixes and suffixes selected. For example, it may be the case that `␣zi` is a particularly characteristic sequence in Dutch, and `␣zi` preceded by `n` is more characteristic than `␣zi` followed by any other letter. Due to the constraint of only using 4-byte sequences, we do not consider `␣zi` as a possible feature. However, as we saw in Section 5.5.5, the overall tendency is for a cross-domain classifier to be more accurate if using byte 4-grams rather than byte $n$-grams. There are some exceptions to this, as we saw on a per-dataset level in Table 5.12, and we will examine this in more detail in our error analysis in Section 5.6.3.

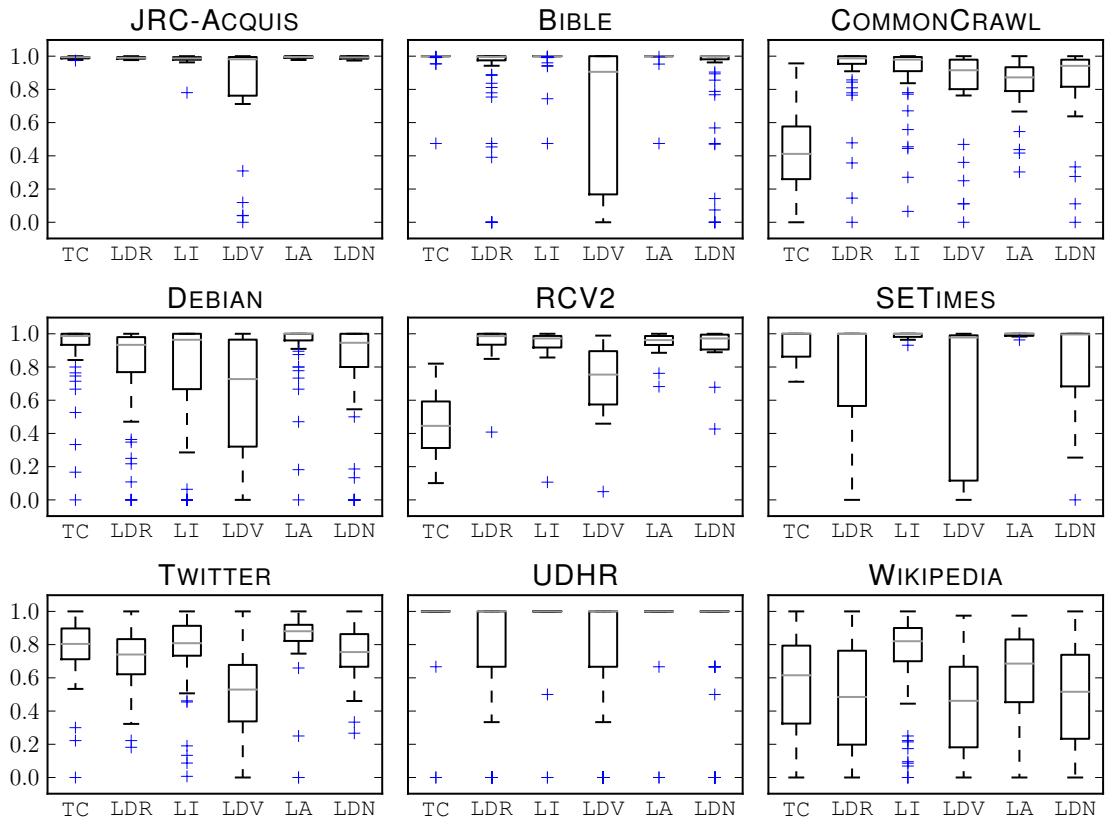



Figure 5.8: Per-dataset boxplot of distribution of F-Scores over the languages in each dataset, broken down by classifier. Each column corresponds to one classifier and are labeled as follows: `TC`=`TextCat`, `LI`=`Linguini` `LA`=`LangDetect`, `LDR`=RANKLISTMODEL using $\mathbb{LD}$ features, `LDV`=VECTORSPACEMODEL using $\mathbb{LD}$ features, `LDN`=LIKELIHOODMODEL using $\mathbb{LD}$ features.

## 5.6 Error Analysis

So far in this chapter, we have succeeded in developing a document representation based on feature selection that takes into account information about the relationship between features (as sequences of 4 consecutive bytes, aka byte 4-grams), the lan-

guage the document is written in and the text source the document comes from. We demonstrated the utility of such a representation in building language identifiers that are more robust to variation in language across sources of text than well-know existing systems. However, the results also indicate that the problem of source-independent LangID, an aspect of generalized LangID, is far from solved. Figure 5.8 presents a comparison of the distribution of per-language F-score between each learning algorithm trained on *out-of-domain* data using cross-domain feature selection (the LD$x$ columns – which are replicated from Figure 5.4), and the 3 standard systems trained on *in-domain* data (a breakdown of the results from Table 4.4). We see that results over DEBIAN, TWITTER and WIKIPEDIA tend to be generally better when using in-domain training data. In this section, we examine selected results in greater detail, with the aim of gaining insight into the outstanding issues which may guide future work on this area. Unless otherwise mentioned, we focus our error analysis on the cross-domain output for a classifier trained using $\mathbb{LD}$ feature selection with the LIKELIHOODMODEL algorithm, selecting 200 features per language, which is one of the best combinations that we identified in the previous section.

### 5.6.1 Number of External Domains

One factor that varies per-language in our experiments is the number of different sources of text from which we have labeled documents. Not all the sources cover all the languages; in fact, due to the inclusion of the SETIMES dataset and the absence of English from RCV2 there is no single language that is present in all the sources we identified in Chapter 3. In Section 4.4.1, we identified the 5 languages that are present

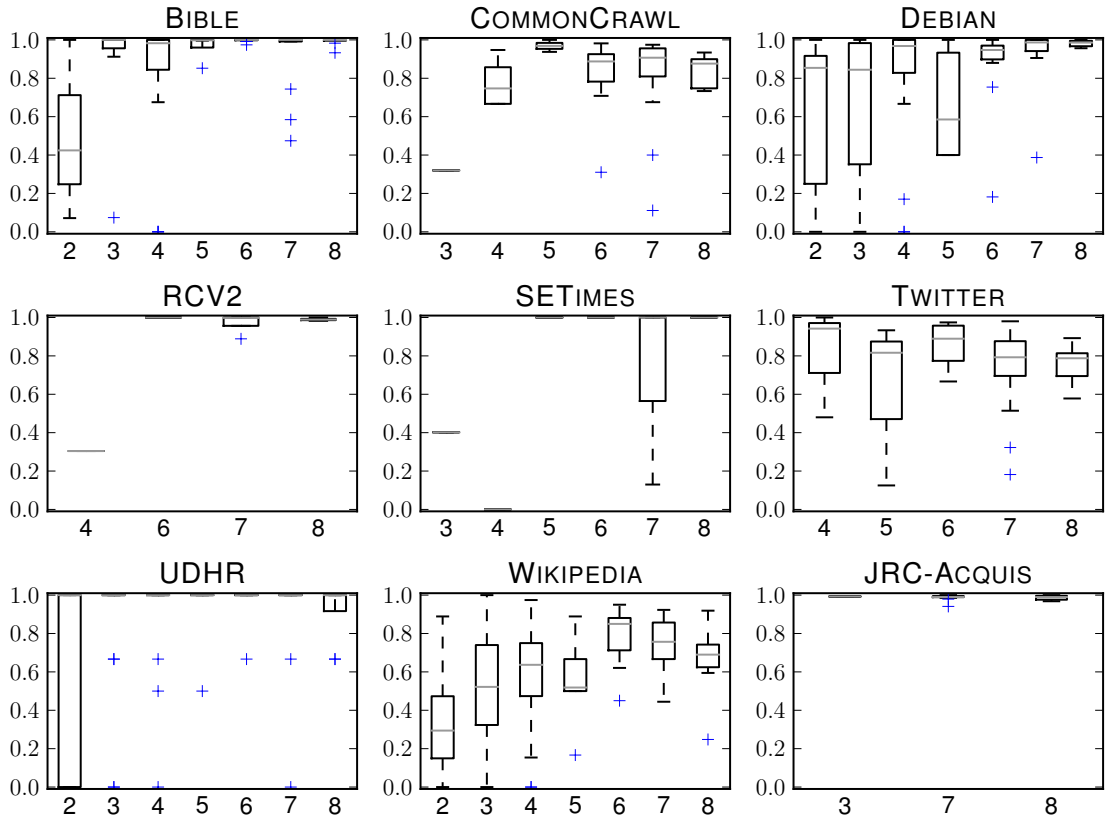



Figure 5.9: Boxplot of macro-averaged F-score per-language broken down by number of text sources that the language is present in. The results presented are from the use of the LIKELIHOODMODEL algorithm with $\mathbb{LD}$ feature selection.

in all the datasets except SETIMES, namely Swedish, Italian, French, Spanish and Danish. The remaining 140 languages are present in at least 2 different sources, but the number of sources varies per-language. Figure 5.9 presents a breakdown of one of the consistently best-performing combinations, the use of $\mathbb{LD}$ feature selection with the LIKELIHOODMODEL algorithm. In this breakdown, languages are grouped by the number of different sources in which they are present. From this plot, we can see that there generally is a relationship between how many sources a language is present in

and how accurate the system is for that language. In particular, where a language is only present in 2 sources, it means that for each target dataset the training data is only drawn from a single source. For some languages, this works reasonably well (Bible, Debian and Wikipedia all contain languages that have been identified with high accuracy using only training data from one other source), however for others it completely fails (again Bible, Debian and Wikipedia all contain languages that only have training data from one other source, where the accuracy on the target dataset is very poor). The general trend is that the more different sources we have, the narrower the range of our expected performance; this is particularly evident in Wikipedia, and to a lesser extent on Bible and Debian. However, there also appears to be a plateau effect; it seems that generally, having more than 4 sources of data for a language (i.e. 3 sources of training data in addition to the test data) does not provide any further benefit to accuracy.

### 5.6.2 Number of Features Selected Per-language

One of the parameters in $\mathbb{LD}$ feature selection is the number of features per-language. We have implemented this as a global parameter, selecting the same number of features per-language, similarly to Cavnar and Trenkle (1994). An alternative approach would be to consider a different number of features per-language, however based on an analysis of our existing results, it seems that this would not yield much benefit. For most languages, there is almost no difference in results between selecting 50 features per-language and selecting 300. In Figure 5.10, we display a per-language breakdown of F-score across the 9 different datasets (noting that not every language

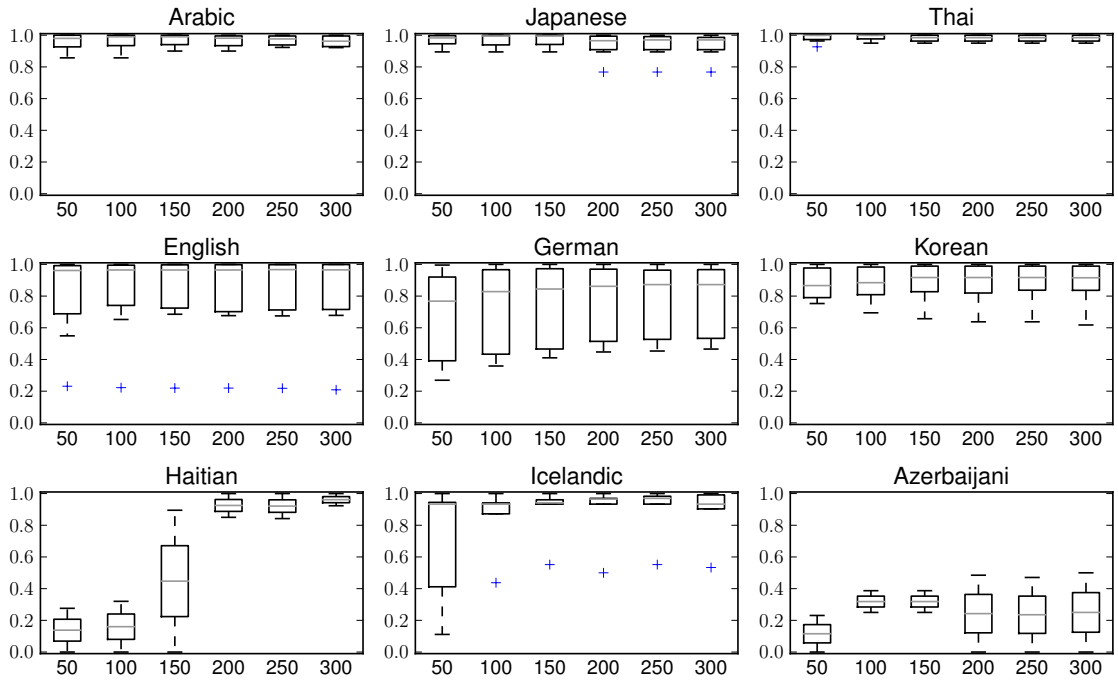



Figure 5.10: Boxplot of distribution of F-score per-language across datasets, broken down by number of features selected per-language using $\mathbb{LD}$.

is present in every dataset). The languages displayed have been selected to illustrate 3 particular trends. In the first row are languages where accuracy is high in all the datasets we have tested. Unfortunately, this is a minority of languages. Another minority is languages in which varying the number of features has a large effect on the accuracy. Examples of such languages are presented in the third row, which shows that the classifier for Haitian trained with 200 features per language is much more accurate than the classifier trained with 100 features per language. Icelandic is slightly different, with a much wider range of results when selecting only 50 features per language, which narrows considerably when using 150 features per language. Finally, Azerbaijani is the only example of a language where increasing the number

of features actually decreases the overall accuracy. This is however an isolated case.

Most languages exhibit a pattern similar to the second row, where accuracy is high in some datasets and much lower in others, though the number of features selected per-language has little to no effect on this. German is perhaps a slight exception, where an increasing number of features does have some positive effect on the average F-score attained across the datasets where German is present. The result for Korean is perhaps somewhat surprising since like Thai, Korean has a distinct orthography that is not shared with any other language. We will see in Section 5.6.3 that the poor performance for Korean is due to an interaction between the encoding and the use of a byte 4-gram tokenization.

### 5.6.3 Byte 4-gram vs $n$-gram

In Section 5.5.5, we observed that tokenization based on byte 4-grams yielded classifiers that were generally more accurate than ones based on byte $n$-grams, though the absolute difference was small and the optimal tokenization varied by text source. Figure 5.11 provides a visualization of this result in the form of a scatter plot for each text source. Each point in the plot represents a single language. We see that for Twitter, there is a tendency towards better performance using $n$-gram features. This can likely be attributed to the short length of Twitter messages. In this case, the shorter $n$-gram features are more likely to occur, and thus the accuracy on Twitter is better because the 4-gram features are relatively undersampled. Debian and Wikipedia paint a more balanced picture, with similar numbers of languages seeing better and worse results, making the overall macro-averaged F-score comparable.

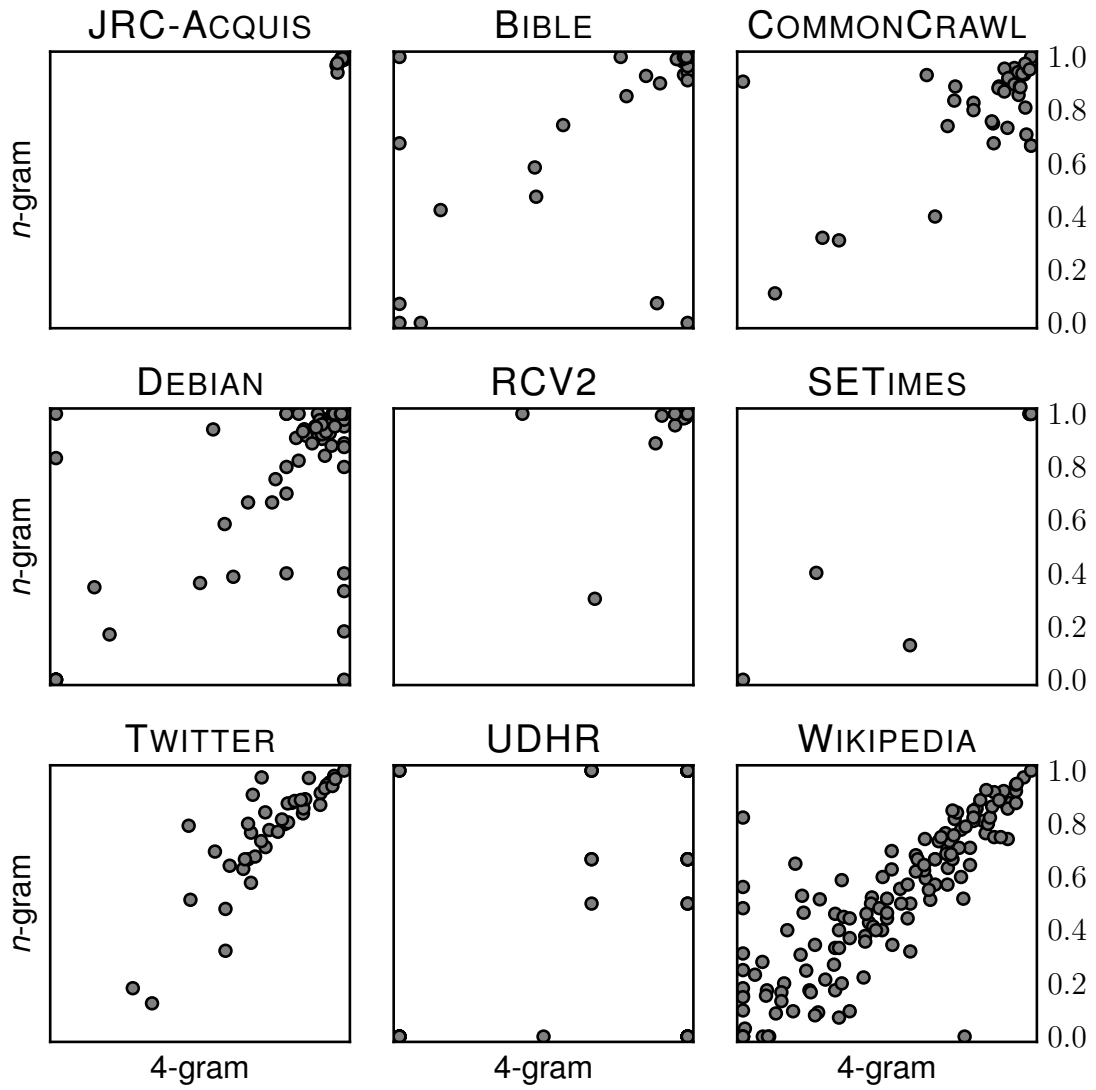



Figure 5.11: Comparison of per-language F-score between byte 4-gram and byte $n$-gram tokenization, broken down by target dataset. In each case, 200 features are selected by $\mathbb{LD}$ and LikelihoodModel is used to train a classifier.

Figure 5.12 presents a subset of the same results used to produce Figure 5.11, grouped by language instead of by dataset. The top row contains examples of languages where the $n$-gram tokenization produces better results, and the bottom row

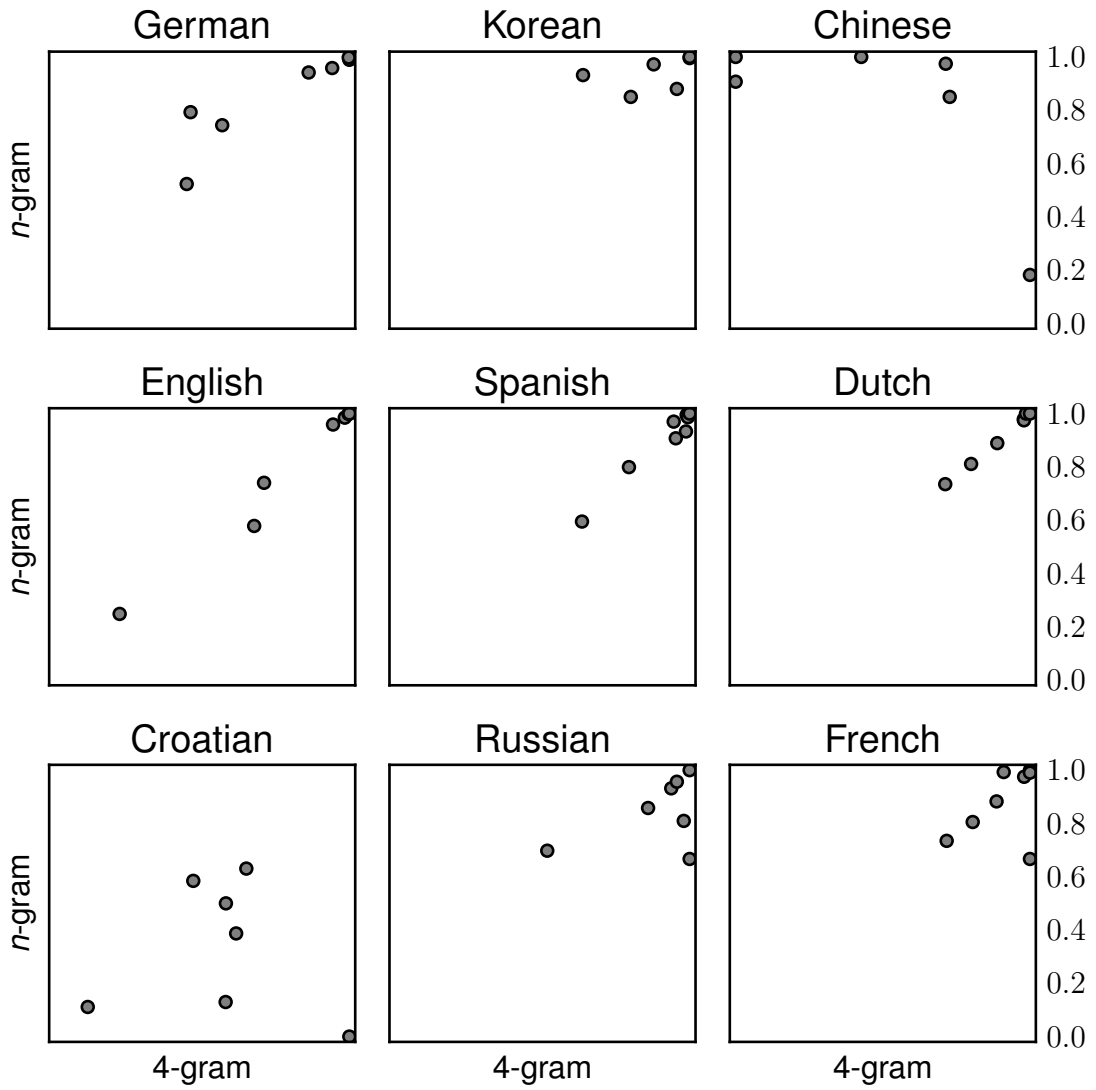



Figure 5.12: Comparison of per-dataset F-score between byte 4-gram and byte $n$-gram tokenization, broken down by language. In each case, 200 features are selected by $\mathbb{LD}$ and LikelihoodModel is used to train a classifier.

contains examples of languages where the 4-gram tokenization is better. The middle row is interesting because it shows that for some of the higher-density languages, the tokenization used does not matter. What is particularly unusual is that in some cases,

very similar less-than-perfect accuracy is attained by models using the two different tokenizations. This suggests that the same documents are being misclassified regardless of the tokenization used. This could be due to the goldstandard labels being incorrect, or due to particular confusability of one language for another. We look at both of these issues in more detail in Section 5.6.5. Of the languages where the byte $n$-gram representation performs better, the result is particularly extreme for Chinese. This is likely due to issues with multi-byte encodings used for Chinese, which can vary in length from 2 bytes (e.g. in GB2312) to 4-bytes (e.g. in UTF8). Enforcing a 4-byte tokenization results in less predictive sequences for Chinese, which impacts the recall of the classifier – under a 4-byte tokenization, many Chinese documents are labeled with other languages. This problem largely disappears when we switch to an $n$-gram tokenization. What is likely happening is that due to the predominance of UTF8 encoding in our datasets, we find that the $n$-gram tokenization is identifying codepoint fragments that are strongly predictive of Chinese – fragments corresponding to the upper byte of the codepoint, which indicates the codeplane that the codepoint is from. We note that using $n$-gram tokenization, there is one particularly poor result in Chinese. This is the result for DEBIAN, where we have very poor precision – documents from many other languages are being labeled as Chinese. This is again likely due to encoding issues, as DEBIAN has a large variety of encodings, whereas most of our other datasets are encoded in UTF8. Due to the density of the GuoBiao and Big5 encodings for Chinese, which are present in COMMONCRAWL, the only dataset with multiple encodings, Chinese ends up as a "catch-all" for unfamiliar encodings as it is the most likely language to contain any arbitrary byte sequence. This illus-

trates a particular challenge in dealing with language-specific encodings that have no guarantee of mutual exclusivity, and is an aspect that could be explored in future work. Korean is similar to Chinese in that it uses a multi-byte encoding. As we will see later in Figure 5.13, the main issue with Korean is recall – i.e. Korean documents tend to be misclassified to other languages. Again, this problem is substantially reduced when using an $n$-gram tokenization instead. Overall, one point that we can take away from this analysis is that when multi-byte encodings are involved, byte $n$-gram tokenization is preferable to byte 4-gram tokenization as it allows us to exploit structure in encodings to identify scripts through the sharing of codepoint fragments that determine the codeplane that the codepoint is from.

### 5.6.4 Precision vs Recall

Figure 5.13 explores the relationship between precision and recall over a selection of 9 languages. The languages have been selected to highlight 3 general patterns. In the top row, we have languages that tend to suffer in terms of precision. In the second row, we have languages that tend to suffer in terms of recall, and in the third row we have languages that variously suffer in both precision and recall.

The languages that have poor precision (top row of Figure 5.13) tend to be "major" languages, i.e. languages that have a large number of training documents. Poor precision indicates that many documents from other languages are being classified into these languages. This can occur for a number of reasons. The lowest precision in English is obtained in WIKIPEDIA, where precision is just 12.5%. Manual inspection of the mislabeled documents however reveals that the problem is not in the classifier, but

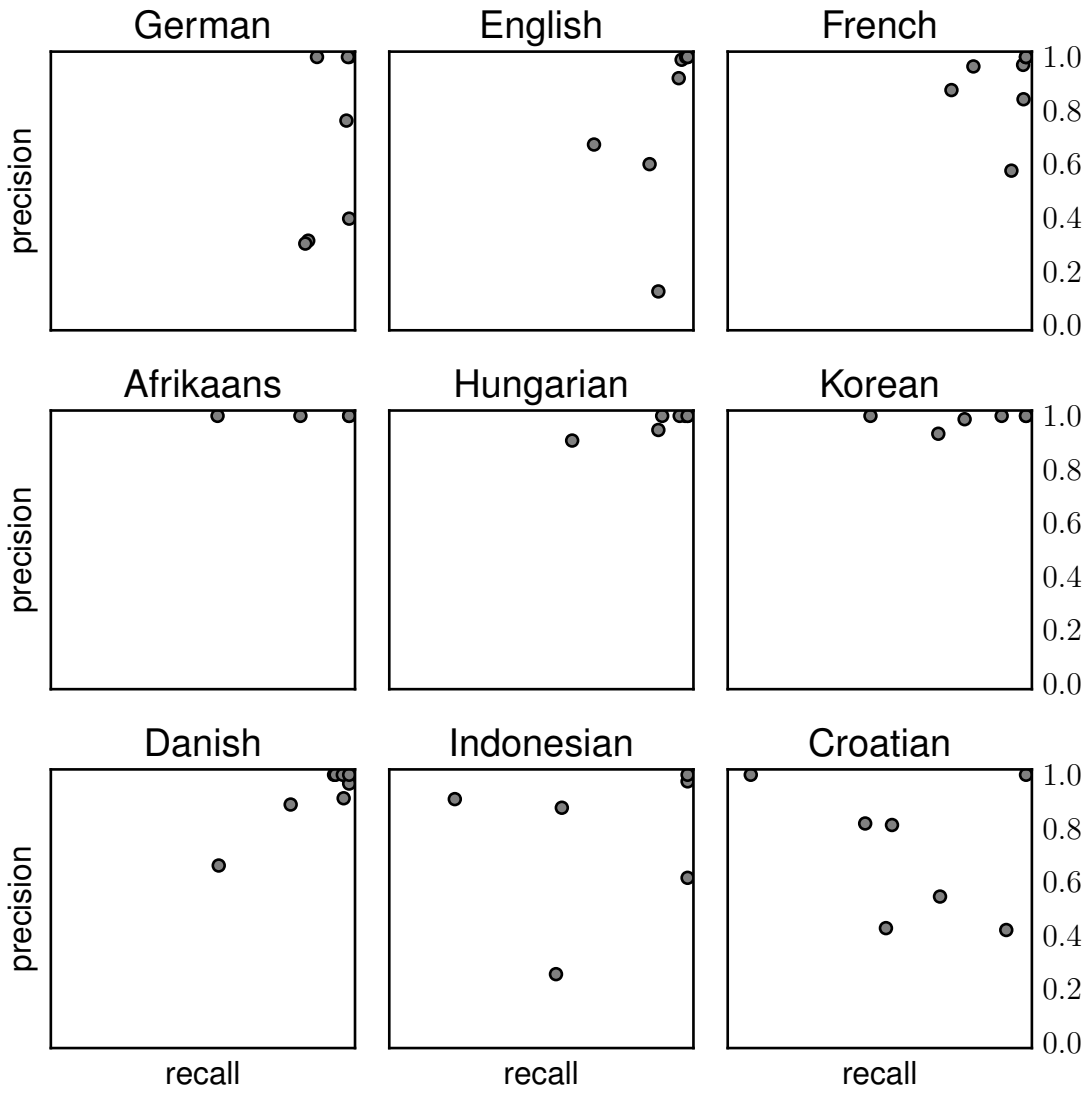



Figure 5.13: Scatter plot of precision vs recall on a per-language basis for a selected subset of languages. Each point is the precision and recall on a single dataset, using $\mathbb{LD}$ feature selection and the LikelihoodModel algorithm. 200 features were selected per-language.

rather in the dataset itself. In constructing the dataset for Wikipedia (Section 3.2.5), we assumed that all documents in a language-specific Wikipedia were written in that

language. However, of the mislabeled documents, a large number are actually written in English or contain a large proportion of text in English. This can occur in the form of notes, or as sections marked "pending translation". Another common reason for the presence of English-language pages in non-English Wikipedias is the use of template pages that have yet to be filled in by the community. Overall, this indicates that care must be taken in using Wikipedia as a source of training data. These issues could likely be resolved with some dataset-specific preprocessing, however our aim in this thesis was to avoid this as far as possible.

For German, poor precision is observed in the results on BIBLE (39.7%), TWITTER (31.4%) and WIKIPEDIA (30.4%). For the BIBLE dataset, several languages with distinct orthographies are identified entirely as German. In some cases this is due to a flaw of the dataset, for example the documents for Estonian (`et`) in the BIBLE dataset are largely blank. German is the predicted language in this case as it is the majority class in the training data, and the LIKELIHOODMODEL algorithm includes a class size prior. This also explains many other documents that have been misclassified as German, the primary cause being that the document did not contain any features in the feature set, and thus the classification was based entirely on the document prior. As we saw in Section 5.6.3, this can happen for languages with multi-byte encodings, where a fixed-length tokenization can be sub-optimal as it may be the same size or larger than a codepoint in the language, resulting in byte sequences that are unable to generalize characteristics such as the codeplane of the codepoints used. Using an $n$-gram representation for such languages can alleviate the problem, as illustrated by the improved German results using $n$-grams in Figure 5.11.

For French, the RCV2 result presents particularly poor precision due to about two-thirds of Chinese documents being classified as French, the rest being correctly identified as Chinese. This problem is not present when $n$-gram tokenization is used and therefore is likely to be related to the issues we discussed in Section 5.6.3.

For the languages that have poor recall (middle row of Figure 5.13), there is generally one dataset where the recall is especially poor. For Afrikaans, this is WIKIPEDIA. For Hungarian, it is TWITTER and for Korean, it is COMMONCRAWL. We have already examined the Korean result in more detail in Section 5.6.3. For Afrikaans, the main languages that it is being confused with are West Frisian (`fy`) and Tartar (`tt`). West Frisian is a language spoken in the Netherlands, and so it is perhaps not surprising that there may be some overlap with Afrikaans. Tartar is a much odder case, as it uses Cyrillic script and has no obvious connection with Afrikaans. Manual inspection of the results for Tartar in WIKIPEDIA reveals that it has very poor precision, and furthermore documents from a large variety of languages (108) are misclassified as Tartar. This is likely an artifact of the limited training data – outside WIKIPEDIA, Tatar only has training data in UDHR, and is perhaps a vulnerability of the LIKELIHOODMODEL algorithm, which seems to tend to over-predict languages with very little training data, resulting in the poor performance we saw in Figure 5.9. For Hungarian, as is the case with many languages on TWITTER, there are a number of different languages that messages are misclassified to. We look at the issue in greater detail in Section 5.6.5.

Finally, we consider languages where both precision and recall are poor. The examples that we have selected, Danish, Indonesian and Croatian have one aspect in

| | bs | hr | mk | pl | rw | sr | su | tt | yo |
|---|---|---|---|---|---|---|---|---|---|
| bs | 48 | 192 | 0 | 1 | 0 | 0 | 0 | 5 | 0 |
| hr | 139 | 483 | 0 | 1 | 8 | 3 | 8 | 2 | 6 |
| mk | 0 | 0 | 385 | 0 | 0 | 0 | 0 | 1 | 0 |
| pl | 1 | 5 | 0 | 529 | 0 | 1 | 0 | 0 | 1 |
| rw | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 6 | 0 |
| sr | 74 | 354 | 34 | 0 | 0 | 1 | 0 | 0 | 0 |
| su | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 3 | 0 |
| tt | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 |
| yo | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 1 |

Table 5.13: Confusion matrix for Bosnian (bs), Serbian (hr) and Croatian (sr).

common: they all belong to groups of closely-related languages that have previously been reported in the literature as being difficult for LangID. Prager (1999a) identifies Danish and Norwegian Bokmal as being easily confusable, Ranaivo-Malancon (2006) identifies Malay and Indonesian as being closely related and Tiedemann and Ljubešić (2012) identify Bosnian, Serbian and Croatian as being closely related. We investigate this property in more detail in Section 5.6.5.

### 5.6.5 Confusion Matrices

One outstanding issue that has been identified in LangID research to date is discrimination between closely-related languages. As discussed in Section 2.5.6, much recent work has been dedicated to discriminating between specific sets of closely related languages. Previous work has used confusion matrices as a means to examine the distribution of misclassified documents over the languages that they have been misclassified to (see Section 2.2.4), and so we begin our analysis of "confusable" languages by a similar analysis of confusion matrices. We do not present a full

confusion matrix here as it would be impractical to do so over the 145 languages we consider. Instead, we present selected subsets of the matrix in order to illustrate particular groups of easily-confused languages. Unless otherwise noted, we present the matrices as the sum of errors across all the datasets we have tested, to illustrate general trends rather than dataset-specific peculiarities. For each matrix we present, we start with an initial seed set of languages $S$, and construct a second set of languages $T$. A language $l$ is added to $T$ if either (or both) of the following criteria hold: (1) there are at least 5 false negatives from a language in $S$ to $l$, or (2) there are at least 5 false positives from $l$ to a language in $S$. We then present the subset of the confusion matrix which includes all the languages in $S \cup T$.

Table 5.13 presents the confusion matrix for Bosnian (`bs`), Serbian (`hr`) and Croatian (`sr`). Consistent with previous work (Tiedemann and Ljubešić 2012), we find that there is a level of mutual confusability between the languages. In contrast to Tiedemann and Ljubešić (2012), who only considered Bosnian, Serbian and Croatian, by considering all the languages in our dataset we also find that Serbian documents can be occasionally misclassified as Macedonian (`mk`), which is plausible due to the geographic proximity of the majority of the speakers of each language. There is also a tendency in our experiments for some documents in each of these languages to be misclassified as Tartar (`tt`), though we accounted for this as a peculiarity of our experimental setup in Section 5.6.4.

Another pair of languages that are known to be highly similar and difficult to distinguish are Malay (`ms`) and Indonesian (`id`) (Ranaivo-Malancon 2006). Table 5.14 presents the confusion matrix for these two languages, and results are again consistent

| | de | id | jv | ms | su |
|---|---|---|---|---|---|
| de | 1072 | 0 | 0 | 0 | 0 |
| id | 12 | 279 | 15 | 29 | 27 |
| jv | 0 | 12 | 4 | 0 | 0 |
| ms | 0 | 21 | 0 | 8 | 4 |
| su | 0 | 9 | 0 | 0 | 4 |

Table 5.14: Confusion matrix for Malay (ms) and Indonesian (id).

with previous work, with the added caveat that we identify some additional languages that are part of this confusion set: Javanese (jv) and Sundanese (su), which are both South-East Asian languages (the presence of German (de) is due to the majority-class prior, as discussed in Section 5.6.4).

Research to date on closely related languages has generally focused on confusability between languages in a pre-specified set. However, one aspect that has not been explored is how to integrate results from research on a specific set of languages into the broader context of LangID. As mentioned in Section 2.5.6, the DSL Shared Task (Zampieri *et al.* to appear) challenged participants to discriminate between 13 languages organized into 6 groups. Participants were required to identify both the group that a document belonged to, as well as the specific language or variety. The results showed that in this small group of languages, identifying the group correctly is easy with existing techniques, and a two-level classifier that first identifies the group and then separately the variety within the group is an effective method of undertaking the task. However, the feasibility of this approach also depends on our ability to identify sets of languages to model as a group, which were assumed to be known in advance in the shared task.

Figure 5.14 shows a plot of the number of different languages each language is

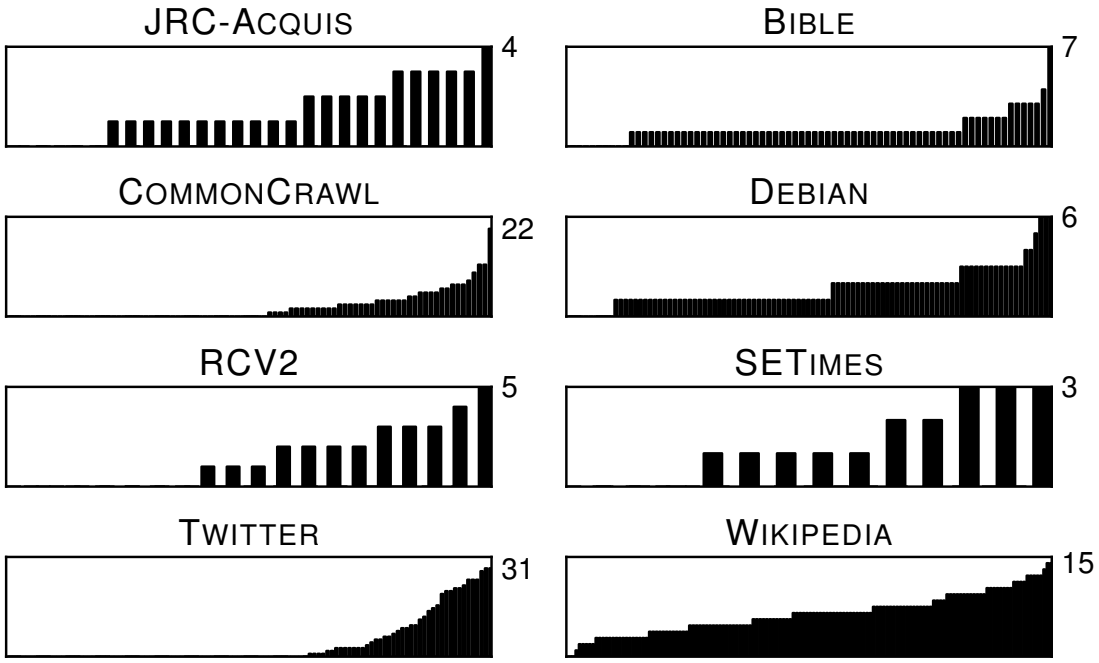



Figure 5.14: Number of different languages that each language is misclassified to, broken down by dataset.

misclassified into, broken down by dataset. UDHR is excluded from this figure because there is only one test document per language. We find that generally, the number of languages to which any given language is misclassified is fairly small, which lends support to the idea that it is possible to first identify a group of languages to which a document belongs, and then in a second step determine the exact language from the group. However, there are some notable outliers. In CommonCrawl, Chinese documents were misclassified into 22 different languages. The reasons for this are related to encoding, and have been discussed in more detail in Section 5.6.3. Twitter is the dataset where documents tend to be misclassified into the largest variety of languages, and this is likely due to the "documents" (i.e. single messages)

being relatively short. However, we do see that there is an overall trend towards documents being misclassified into a relatively small number of languages, particularly bearing in mind that the results in Figure 5.14 are based on an *out-of-domain* classifier that could output any language in the training set, even if it was not present in the test set. Overall, this means that a 2-level hierarchical classifier is promising for improving the overall accuracy of LangID, however work remains to be done on identifying the exact languages that should be treated as a group: as we have seen in Table 5.13 and Table 5.14, the confusion sets identified in work to date are insufficiently broad to capture the full range of languages that are easily confused.

### 5.6.6 Poor accuracy of VectorSpaceModel using $\mathbb{LD}$ features

One anomaly noticeable in Figure 5.4 on page 190 is that the VectorSpaceModel classifier shows unexpectedly high variance between classes when combined with $\mathbb{LD}$ features on data from RCV2 and JRC-Acquis. This is particularly odd because RCV2 and particularly JRC-Acquis can be considered two of the "easier" text sources, having longer documents in a relatively limited number of languages, with limited source-specific "noise". A detailed examination of the predictions for JRC-Acquis shows that recall in Portuguese, Italian, German and Slovak is close to 0. This is due to documents in each of these languages being incorrectly identified as being in one of a small group of other languages. For Portuguese, this was Galician. For Italian, this was Romansh and Corsican. For German, this was Zhuang, for Danish this was Norwegian, and for Slovak this was Czech. With the exception of German, each language is being consistently mis-identified as a closely-related lan-

guage. The exact reasons for this interaction are unclear, but may be due to issues of multi-modality (in the statistical sense of mode). One possible explanation is that in the VectorSpaceModel, documents from each language are still forming distinct clusters for each source (despite our efforts to only model source-independent features), with related languages appearing "between" clusters for each language. Since in the experiments reported in Figure 5.4, no data from the test source is used in training, it is possible that the test source falls between existing clusters for the language, and is thus closer to one of the intervening closely-related languages than to any of the clusters from the "correct" language.

Another anomaly occurs in Table 5.12 on page 196, where VectorSpaceModel accuracy is substantially lower on $n$-gram representations than 4-gram representations, especially on JRC-Acquis and CommonCrawl, in contrast with the general trend of $n$-gram and 4-gram accuracy being comparable. Detailed analysis of the JRC-Acquis result shows that the errors being made using the $n$-gram and 4-gram representations are quite different. In contrast to the errors we reported in the previous paragraph for the 4-gram representation, in the $n$-gram representation the majority of errors are due to misclassification as English, Polish to Lithuanian, and Portuguese to English, Spanish and Guarani. Despite the actual errors being different, it is likely that the root cause is similar: despite the use of $\mathbb{LD}$ features, the VectorSpaceModel model is still modeling distinct regions of the vector space for documents from each source despite the documents being in the same language, with the possibility of interceding clusters of a different language. Thus, when documents from a different source are classified, they may be identified as "between" clusters for

a language and thus closer to another language. This concept of interceding clusters is consistent with an issue identified by Prager (1999a), where mixed-language documents appear "between" the two constituent languages and are thus closer to a third language than either constituent in the vector space.

## 5.7 Chapter Summary

In this chapter, our aim was to examine the underlying causes for the difference in performance of well-known LangID systems when training and test data for the same language were drawn from different sources as opposed to the setting where all data comes from the same source, and to develop a strategy to mitigate this loss in accuracy.

We dissected the three systems from Chapter 4 and showed how all three represent documents as a distribution over sequences of letters selected by term frequency, and apply supervised machine learning in order to predict the language of an unseen document. We further broke down the machine learning approach into distinct feature selection and classification stages. In the feature selection stage, we showed how variation in the same language between different text sources is a potential source of bias, and mitigated this bias through a cross-domain feature selection strategy. We then identified relative length of documents in each language as another potential source of bias in LangID, and showed that existing algorithms commonly applied to LangID are already robust to this bias, which may explain why algorithms that fit more complex boundaries are not able to outperform "simpler" learning algorithms for LangID.

We found that the document representation induced by our method is more homogeneous with respect to language across text sources than the document representations used by existing methods, and empirically demonstrated that this improved representation leads to improved accuracy with respect to existing systems regardless of the learning algorithm used when applied in the cross-domain LangID setting. However, we also observed that despite a substantial improvement in the cross-domain setting, our trained classifiers were not able to attain the accuracy of the same algorithms trained on in-domain training data. We carried out an error analysis, which identified some of the main sources of error, amongst which are issues of encoding, sparsity of training data, and closely-related languages. Overall, we achieved our initial objective of mitigating the loss in accuracy when applying a language identifier to out-of-domain data by developing a document representation that is more robust to variation in the same language between different sources of text.

So far however, we have continued to assume that each document contains text from a single language. In the next chapter, we tackle this *monolinguality assumption*, building on the document representation we proposed in this chapter to develop a LangID system that is able to detect when more than one language is present in a document, identify the languages that are present and also estimate the relative proportions of the document written in each language.

# Chapter 6

# Language Identification of Multilingual Documents

Thus far in this thesis, we have examined LangID under the assumption that every document is written in exactly one of a closed set of known languages for which there is training data, and we have thus formulated LangID as the task of selecting the most likely language from the set of training languages. In this chapter, we remove this monolingual assumption, and address the problem of LangID in documents that may contain text from more than one language from the candidate set. We introduce a method that is able to detect multilingual documents, and simultaneously identify each language present as well as estimate the proportion of the document written in

This chapter is based on work previously published as:

Lui, Marco, Jey Han Lau, and Timothy Baldwin. 2014. Automatic Detection and Language Identification of Multilingual Documents. *Transactions of the Association for Computational Linguistics*, 2(Feb):27 – 40.

that language. We achieve this with a probabilistic mixture model, building on the document representation we developed for monolingual LangID in Chapter 5. The model posits that each document is generated as samples from an unknown mixture of languages from the training set. We introduce a Gibbs sampler to map samples to languages for any given set of languages, and use this to select the set of languages that maximizes the posterior probability of the document.

Our method is able to learn a language identifier for multilingual documents from monolingual training data. This is an important property as there are no standard corpora of multilingual documents available, whereas corpora of monolingual documents are readily available for a reasonably large number of languages, as we saw in Chapter 3. We demonstrate the effectiveness of our method empirically, firstly by evaluating it on synthetic datasets drawn from Wikipedia data, and then by applying it to real-world data, showing that we are able to identify multilingual documents in targeted web crawls of minority languages (King and Abney 2013).

## 6.1 Multi-label Classification

Multi-label classification is the task of assigning zero or more labels $c_i$ from a class set $C$ to an instance (Tsoumakas and Katakis 2007). Multilingual LangID is thus a multi-label classification problem, where we will follow convention in maintaining the closed-world assumption: i.e. assuming that a given test document contains at least one of the languages in the training data.

Some models, such as instance-based learners, support multi-label classification directly, in returning a set of $k$ training instances of greatest similarity to the test

instance, amongst which we can simply take the union of the class labels. Alternatively, for classifiers which return a score (or probability) per class, a threshold over the scores can be used to generate multi-label outputs (Schapire and Singer 2000; Gao *et al.* 2004).

More sophisticated approaches to multi-label classification attempt to explicitly capture dependencies between labels, e.g. via conditional random fields (Ghamrawi and McCallum 2005) or generative mixture models (McCallum 1999; Ueda and Saito 2002). The model used in this chapter is a generative mixture model, similar in some ways to supervised variants of Latent Dirichlet Allocation (Blei *et al.* 2003; Griffiths and Steyvers 2004; Ramage *et al.* 2009). We discuss this relationship further in Section 6.2.2.

## 6.2 Methodology

Language identification for multilingual documents is a multi-label classification task, in which a document can be mapped onto any number of labels from a closed set. In the remainder of this chapter, we denote the set of all languages by $L$. We denote a document $D$ which contains languages $L_x$ and $L_y$ as $D \rightarrow \{L_x, L_y\}$, where $L_x, L_y \in L$. We denote a document that does not contain a language $L_x$ by $D \rightarrow \{\overline{L_x}\}$, though we generally omit all the languages not contained in the document for brevity. We denote classifier output using $\triangleright$; e.g. $D \triangleright \{L_a, L_b\}$ indicates that document $D$ has been predicted to contain text in languages $L_a$ and $L_b$.

| | **English** | **French** | **Italian** | **German** | **Dutch** | **Japanese** |
|---|---|---|---|---|---|---|
| character | the␣ | pour | ␣di␣ | ␣auf | voo | は |
| byte | 74 68 65 20 | 70 6F 75 7 | 20 64 69 20 | 20 61 75 66 | 76 6F 6 | E3 81 AF |

Table 6.1: Examples of per-language byte sequences selected by information gain.

### 6.2.1 Document Representation and Feature Selection

Our representation for each document $D$ is a frequency distribution over byte $n$-gram sequences (examples are given in Table 6.1), which is conceptually the same as the representation developed in Chapter 5. Each document is converted into a vector where each entry counts the number of times a particular byte $n$-gram is present in the document. As discussed in Section 2.2.1, this is analogous to a bag-of-words model, where the vocabulary of "words" is a set of byte sequences that has been selected to distinguish between languages.

The exact set of features is selected from the training data using Information Gain (IG). This is closely related to the approach presented in Chapter 5, the main difference being that in this chapter we begin our investigation by only considering in-domain training data, and so the method developed in Chapter 5 does not immediately apply. We revisit the issue of cross-domain training at the end of this chapter (Section 6.6.1).

### 6.2.2 Generative Mixture Models

Generative mixture models are popular for text modeling tasks where a mixture of influences governs the content of a document, such as in multi-label document classification (McCallum 1999; Ramage *et al.* 2009), and topic modeling (Blei *et al.*

2003). Such models normally assume full exchangeability between tokens (i.e. the bag-of-words assumption), and label each token with a single discrete label.

Multi-label text classification, topic modeling and our model for LangID in multilingual documents share the same fundamental representation of the latent structure of a document. Each label is modeled with a probability distribution over tokens, and each document is modeled as a probabilistic mixture of labels. As presented in Griffiths and Steyvers (2004), the probability of the $i^{th}$ token ($w_i$) given a set of $T$ labels $z_1 \cdots z_T$ is modeled as:

$$P(w_i) = \sum_{j=1}^{T} P(w_i|z_i = j)P(z_i = j) \tag{6.1}$$

The set of tokens $w$ is the document itself, which in all cases is observed. In the case of topic modeling, the tokens are words and the labels are topics, and $z$ is latent. Whereas topic modeling is generally unsupervised, multi-label text classification is a supervised text modeling task, where the labels are a set of pre-defined categories (such as RUBBER, IRON-STEEL, TRADE, etc. in the popular Reuters-21578 dataset (Lewis 1997)), and the tokens are individual words in documents. $z$ is still latent, but constrained in the training data (i.e. documents are labeled but the individual words are not). Some approaches to labeling unseen documents require that $z$ for the training data be inferred, and methods for doing this include an application of the Expectation-Maximization (EM) algorithm (McCallum 1999) and Labeled LDA (Ramage *et al.* 2009).

The model that we propose for LangID in multilingual documents is similar to multi-label text classification. In the framework of Equation 6.1, each per-token label $z_i$ is a language and the vocabulary of tokens is not given by words but rather

by specific byte sequences (Section 6.2.1). The key difference with multi-label text classification is that we use monolingual (i.e. mono-label) training data. Hence, $z$ is effectively observed for the training data (since all tokens must share the same label). To infer $z$ for unlabeled documents, we utilize a Gibbs sampler, closely related to that proposed by Griffiths and Steyvers (2004) for LDA. The sampling probability for a label $z_i$ for token $w$ in a document $d$ is given by:

$$P(z_i = j|z_{-i}, w) \propto \phi_j^{(w)} \cdot \theta_j^{(d)} \tag{6.2}$$

$$\phi_j^{(w)} = P(w_i|z_i = j, z_{-i}, w_{-i})$$

$$\theta_j^{(d)} = P(z_i = j|z_{-i})$$

In the LDA model, $\theta_j^{(d)}$ is assumed to have a Dirichlet distribution with hyperparameter $\alpha$, and the word distribution for each topic $\phi_j^{(w)}$ is also assumed to have a Dirichlet distribution with hyperparameter $\beta$. Griffiths (2002) describes a generative model for LDA where both $\phi_j^{(w)}$ and $\theta_j^{(d)}$ are inferred from the output of a Gibbs sampler. In our method, we estimate $\phi_j^{(w)}$ using maximum likelihood estimation (MLE) from the training data. Estimating $\phi_j^{(w)}$ through MLE is equivalent to a multinomial Naive Bayes model (McCallum and Nigam 1998):

$$\hat{\phi}_j^{(w)} = \frac{n_j^{(w)} + \beta}{n_j^{(.)} + W\beta} \tag{6.3}$$

where $n_j^{(w)}$ is the number of times word $w$ occurs with label $j$, and $n_j^{(.)}$ is the total number of words that occur with label $j$. By setting $\beta$ to 1, we obtain standard Laplacian smoothing. Hence, only $\hat{\theta}_j^{(d)}$ is updated at each step in the Gibbs sampler.

The update equation is given by:

$$\hat{\theta}_j^{(d)} = \frac{n_{-i,j}^{(d)} + \alpha}{n_{-i}^{(d)} + T\alpha} \tag{6.4}$$

where $n_{-i,j}^{(d)}$ is the number of tokens in document $d$ that are currently mapped to language $j$, and $n_{-i}^{(d)}$ is the total number of tokens in document $d$. In both cases, the current assignment of $z_i$ is excluded from the count. $T$ is the number of languages (i.e. the size of the label set). For simplicity, we set $\alpha$ to 0. Strictly speaking, this breaks the Dirichlet assumption from LDA since the Dirichlet distribution is only defined for $\alpha > 0$. However, setting $\alpha$ to 0 implies that terms not assigned to a specific language in some document in the training data will never be assigned to that language at classification time. We note that in the LDA model, $\alpha$ and $\beta$ influence the sparsity of the solution, and so it may be possible to tune these parameters for our model as well. We leave this as an avenue for further research.

### 6.2.3 Language Identification in Multilingual Documents

The model described in Section 6.2.2 can be used to compute the most likely distribution to have generated an unlabeled document over a given set of languages for which we have monolingual training data. We do this by letting the set of terms $w$ be the byte $n$-gram sequences we selected using per-language information gain (Section 6.2.1), and allowing the labels $z$ to range over the set of all languages $L$. Using our training data, we compute $\hat{\phi}_j^{(w)}$ (Equation 6.3), and then we infer $P(L_j|D)$ for each $L_j \in L$ for the unlabeled document. This is done by running the Gibbs sampler until the samples for $z_i$ converge, and then tabulating $z_i$ over the whole $d$ and normalizing by $|d|$. Naively, we could then identify the languages present in

**Algorithm 1** *DetectLang*$(L, D)$

$L_N \leftarrow$ top-$N$ $z \in L$ by $P(z|D)$

$\lambda \leftarrow \{L_u\}$

**for each** $L_t \in L_N$ **do**

$\lambda' \leftarrow \lambda \cup L_t$

**if** $P(D|\lambda) + t < P(D|\lambda')$ **then**

$\lambda \leftarrow \lambda'$

**end if**

**end for**

$\lambda \leftarrow \lambda \setminus \{L_u\}$

**return** $D \triangleright \lambda$

the document by $D \triangleright \{L_x \text{ if } \exists(z_i = L_x|D)\}$. The main issue with this approach is that closely-related languages tend to have similar frequency distributions over byte $n$-gram features (see Section 2.5.6), hence it is likely that some tokens will be incorrectly mapped to a language that is similar to the "correct" language.

We address this issue by finding the subset of languages $\lambda$ from the training set $L$ that maximizes $P(\lambda|D)$ (a similar approach is taken in McCallum (1999)). Through an application of Bayes' theorem, $P(\lambda|D) \propto P(D|\lambda) \cdot P(\lambda)$, noting that $P(D)$ is a normalizing constant and can be dropped. We assume that $P(\lambda)$ is constant (i.e. any subset of languages is equally likely, a reasonable assumption in the absence of other evidence), and hence seek to maximize $P(D|\lambda)$. For any given $D = w_1 \cdots w_n$ and $\lambda$,

we infer $P(D|\lambda)$ from the output of the Gibbs sampler as follows:

$$P(D|\lambda) = \prod_{i=1}^{N} P(w_i|\lambda) \tag{6.5}$$

$$= \prod_{i=1}^{N} \sum_{j \in \lambda} P(w_i|z_i = j)P(z_i = j) \tag{6.6}$$

The main difference with the full mixture model is that $\lambda$ constrains certain mixture components to 0 so that they do not contribute to the summation, and so this is equivalent to setting $P(z_i = j) = 0$ for $j$ not in $\lambda$. Both $P(w_i|z_i = j)$ and $P(z_i = j)$ are estimated by their respective maximum likelihood estimates.

In practice, exhaustive evaluation of the powerset of $L$ is prohibitively expensive, and so we greedily approximate the optimal $\lambda$ using Algorithm 1. In essence, we initially rank all the candidate languages by computing the most likely distribution over the full set of candidate languages. Then, for each of the top-$N$ languages in turn, we consider whether to add it to $\lambda$. $\lambda$ is initialized with $L_u$, a dummy language with a uniform distribution over terms (i.e. $P(w|L_u) = \frac{1}{|w|}$). A language is added if it improves $P(D|\lambda)$ by at least $t$. The threshold $t$ is required to suppress the addition of spurious classes. Adding languages gives the model additional freedom to fit parameters, and so will generally increase $P(D|\lambda)$. In the limit case, adding a completely irrelevant language will result in no tokens being mapped to a language, and so the model will be no worse than without the language. The threshold $t$ is thus used to control "how much" improvement is required before including the new language in $\lambda$.

### 6.2.4 Benchmark Approaches

We compare our approach to two other methods for LangID in multilingual documents: (1) the *virtual mixed languages* approach proposed by Prager (1999a); and (2) the text segmentation approach proposed by Yamaguchi and Tanaka-Ishii (2012).

Prager (1999a) describes `Linguini`, a language identifier based on a vector-space model, a commonly used representation in text classification and information retrieval. `Linguini` is one of the systems we evaluated for LangID of monolingual documents in Chapter 4. The document representation used by Prager (1999a) is a vector of counts across a set of character sequences. Prager (1999a) selects the feature set based on a metric in the style of the Term Frequency – Inverse Document Frequency (TF – IDF) family of metrics used for information retrieval. Terms with occurrence count $m < n \times k$ are rejected, where $m$ is the number of times the term occurs in the training data (the TF component), $n$ is the number of languages in which the term occurred (the IDF component, where "document" is replaced with "language"), and $k$ is a parameter to control the overall number of terms selected. In Prager (1999a), the value of $k$ is reported to be optimal in the region 0.3 to 0.5. In practice, the value of $k$ indirectly controls the number of features selected. Values of $k$ are not comparable across datasets as $m$ is not normalized for the size of the training data, so in this work we do not report the values of $k$ and instead directly select the top-$N$ features, weighted by $\frac{m}{n}$.

In `Linguini`, each language is modeled as a single pseudo-document, obtained by concatenating all the training data for the given language. A document is then classified according to the vector with which it has the smallest angle; this is implemented

by finding the language vector with the highest cosine with the document vector.

Prager (1999a) also proposes an extension to the approach to allow identification of bilingual documents, and suggests how this may be generalized to any number of languages in a document. The gist of the method is simple: for any given pair of languages, the projection of a document vector onto the hyperplane containing the language vectors of the two languages gives the mixture proportions of the two languages that minimizes the angle with the document vector. Prager (1999a) terms this projection a *virtual mixed language* (VML), and shows how to find the angle between the document vector and the VML. If this angle is less than that between the document vector and any individual language vector, the document is labeled as bilingual in the two languages from which the mixed vector was derived. The practical difficulty presented by this approach is that exhaustively evaluating all possible combinations of languages is prohibitively expensive. Prager (1999a) addresses this by arguing that in multilingual documents, "the individual component languages will be close to $d$ (the document vector) – probably closer than most or all other languages". Hence, language mixtures are only considered for combinations of the top $m$ languages.

Prager (1999a) shows how to obtain the mixture coefficients for bilingual VMLs, arguing that the process generalizes. Prager (1999b) includes the coefficients for 3-language VMLs, which are much more complex than the 2-language variants. Using a computer algebra system, we verified the analytic forms of the coefficients in the 3-language VML. We also attempted to obtain an analytic form for the coefficients in a 4-language VML, but these were too complex for the computer algebra system to

compute. Thus, our evaluation of the VML approach proposed by Prager (1999a) is limited to 3-language VMLs. Neither Prager (1999a) nor Prager (1999b) include an empirical evaluation over multilingual documents, so one of the contributions of this chapter is such an empirical evaluation of the method on multilingual documents. As no reference implementation of this method is available, we have produced our own implementation, which we have made freely available.[1]

The other benchmark we consider in this chapter is the method for text segmentation by language proposed by Yamaguchi and Tanaka-Ishii (2012) (hereafter referred to as `SegLang`). The actual task addressed by Yamaguchi and Tanaka-Ishii (2012) is to divide a document into monolingual segments. This is formulated as the task of segmenting a document $D = x_1, \cdots, x_{|D|}$ (where $x_i$ denotes the $i^{th}$ character of $D$ and $|D|$ is the length of the document) by finding a list of boundaries $B = [B_1, \cdots, B_{|B|}]$ where each $B_i$ indicates the location of a language boundary as an offset from the start of the document, resulting in a list of segments $X = [X_0, \cdots, X_{|B|}]$. For each segment $X_i$, the system predicts $L_i$, the language associated with the segment, producing a list of labellings $L = [L_0, \cdots, L_{|B|}]$, with the constraint that adjacent elements in $L$ must differ. Yamaguchi and Tanaka-Ishii (2012) solve the problem of determining $X$ and $L$ for an unlabeled text using a method based on minimum description length. They present a dynamic programming solution to this problem, and analyze a number of parameters that affect the overall accuracy of the system. Given this method to determine $X$ and $L$, it is then trivial to label an unlabeled document according to $D \triangleright \{L_x \text{ if } \exists L_x \in L\}$, and the length of each segment in $X$ can then be used to

[1] `https://github.com/saffsd/linguini.py`

determine the proportions of the document that are in each language. In this work, we use a reference implementation of `SegLang` kindly provided to us by the authors.

Using the text segmentation approach of `SegLang` to detect multilingual documents differs from `Linguini` and our method primarily in that `Linguini` and our method fragment the document into small sequences of bytes, and discard information about the relative order of the fragments, keeping only the frequency count. This is in contrast to `SegLang`, where this information is utilized in the sequential prediction of labels for consecutive segments of text, and is thus able to make better use of the locality of text (since there are likely to be monolingual blocks of text in any given multilingual document). The disadvantage of this is that the underlying model becomes more complex and hence more computationally expensive, as we observe in Section 6.4.

### 6.2.5 Evaluation

We seek to evaluate the ability of each method: (1) to correctly identify the language(s) present in each test document; and (2) to estimate the relative proportion of the document written in each language. The latter evaluation is carried out over all test documents, regardless of whether the actual test document is monolingual or multilingual, as a method may incorrectly detect a monolingual document as multilingual. In such cases, the evaluation would penalize the method according to the proportion of the document that is incorrectly detected.

Identifying the languages(s) present in each test document is a classification problem, and we evaluate using the standard notions of precision ($\mathcal{P}$), recall ($\mathcal{R}$) and F-

score ($\mathcal{F}$), which we discussed in detail in Section 2.2.4. We report both the document-level *micro-average*, as well as the language-level *macro-average*. The macro-averaged F-score we report is the average of the per-class F-scores, rather than the harmonic mean of the macro-averaged precision and recall; we discussed the implications of this in Section 2.2.4. In Section 2.2.4, we also discussed how in previous work on LangID, the parameter $\beta$ is normally set to 1, giving equal importance to precision and recall. We follow this practice and set $\beta = 1$. Because of the multi-label nature of the task and variable number of labels assigned to a given document by our models, it is theoretically possible and indeed common in our results for the maximal macro-averaged F-score to be achieved when macro-averaged precision and recall are not balanced.

We tested the difference in performance for statistical significance using an approximate randomization procedure (Yeh 2000) with 10000 iterations. Within each table of results (Tables 6.2, 6.3 and 6.4), all differences between systems are statistically significant at a $p < 0.05$ level.

To evaluate the predictions of the relative proportions of a document $D$ written in each detected language $L_i$, we compare the topic proportion predicted by our model to the gold-standard proportion, measured as a byte ratio as follows:

$$gs(L_i|D) = \frac{\text{length of } L_i \text{ portion of } D \text{ in bytes}}{\text{length of } D \text{ in bytes}} \tag{6.7}$$

We report the correlation between predicted and actual proportions in terms of Pearson's $r$ coefficient. We also report the mean absolute error (MAE) over all document – language pairs.

| System | $\mathcal{P}_M$ | $\mathcal{R}_M$ | $\mathcal{F}_M$ | $\mathcal{P}_\mu$ | $\mathcal{R}_\mu$ | $\mathcal{F}_\mu$ |
|---|---|---|---|---|---|---|
| Benchmark | .497 | .467 | .464 | .833 | .826 | .829 |
| Winner | .718 | .703 | .699 | .932 | .931 | .932 |
| `SegLang` | .801 | .810 | .784 | .866 | .946 | .905 |
| `Linguini` | .616 | .535 | .513 | .713 | .688 | .700 |
| Our method | .753 | .771 | .748 | .945 | .922 | .933 |

Table 6.2: Results on the ALTW2010 dataset. "Benchmark" is the benchmark system proposed by the shared task organizers. "Winner" is the highest-$\mathcal{F}_\mu$ system submitted to the shared task.

## 6.3 Experiments on ALTW2010 Dataset

Our first experiment utilizes the ALTW2010 shared task dataset (Baldwin and Lui 2010b), a synthetic dataset of 10000 bilingual documents[2] generated from Wikipedia data, introduced in the ALTW2010 shared task.[3] The dataset is organized into training, development and test partitions. Following standard machine learning practice, we train each system using the training partition, and tune parameters using the development partition. We then report macro and micro-averaged precision, recall and F-score on the test partition, using the tuned parameters.

The results on the ALTW2010 shared task dataset are summarized in Table 6.2. Each of the three systems we compare was re-trained using the training data provided for the shared task, with a slight difference: in the shared task, participants were provided with multilingual training documents, but the systems targeted in this research require monolingual training data. We thus split the training documents

[2]With a small number of monolingual documents, formed by randomly selecting the two languages for a given document independently, leaving the possibility of the same two languages being selected.

[3]`http://comp.mq.edu.au/programming/task_description/`

into monolingual segments using the metadata provided with the dataset. The metadata was only published after completion of the task and was not available to task participants. For comparison, we have included the benchmark results published by the shared task organizers, as well as the score attained by the winning entry (Tran *et al.* 2010).

We tune the parameters for each system using the development partition of the dataset, and report results on the test partition. For `Linguini`, there is a single parameter $k$ to be tuned: the number of features per language. We tested values between 10000 and 50000, and selected 46000 features as the optimal value. For our method, there are two parameters to be tuned: (1) the number of features selected for each language, and (2) the threshold $t$ for including a language. We tested features-per-language counts between 30 and 150, and found that adding features beyond 70 per language had minimal effect. We tested values of the threshold $t$ from 0.01 to 0.15, and found the best value was 0.14. For `SegLang`, we introduce a threshold $t$ on the minimum proportion of a document (measured in bytes) that must be labeled by a language before that language is included in the output set. This was done because our initial experiments indicate that `SegLang` tends to over-produce labels. Using the development data, we found the best value of $t$ was 0.10.

We find that of the three systems tested, two outperform the winning entry to the shared task. This is more evident in the macro-averaged results than in the micro-averaged results. In micro-averaged terms, our method is the best performer, whereas on the macro-average, `SegLang` has the highest F-score. This suggests that our method does well on higher-density languages (relative to the ALTW2010 dataset),

and poorly on lower-density languages. This also accounts for the higher micro-averaged precision but lower micro-averaged recall for our method as compared to `SegLang`. The improved macro-averaged F-score of `SegLang` comes at a much higher computational cost, which increases dramatically as the number of languages is increased. In our testing on a 16-core workstation, `SegLang` took almost 24 hours to process the ALTW2010 shared task test data, compared to 2 minutes for our method and 40 seconds for `Linguini`. As such, `SegLang` is poorly suited to detecting multilingual documents where a large number of candidate languages is considered.

The ALTW2010 dataset is an excellent starting point for this research, but it predominantly contains bilingual documents, making it difficult to assess the ability of systems to distinguish multilingual documents from monolingual ones. Furthermore, we are unable to use it to assess the ability of systems to detect more than 2 languages in a document. To address these shortcomings, we construct a new dataset in a similar vein. The dataset and experiments performed on it are described in the next section.

## 6.4 Experiments on WikipediaMulti Dataset

To fully test the capabilities of our model, we generated WikipediaMulti, a dataset that contains a mixture of monolingual and multilingual documents. To allow for replicability of our results and to facilitate research in LangID, we have made the dataset publicly available.[4] WikipediaMulti is generated using excerpts from the mediawiki sources of Wikipedia pages downloaded from the Wikimedia foundation.[5]

[4] `http://www.csse.unimelb.edu.au/~tim/etc/wikipedia-multi-v5.tgz`

[5] `http://dumps.wikimedia.org`

1. randomly select the number of languages $K$ ($1 \leq K \leq 5$)
2. randomly select a set of $K$ languages $S = \{L_i \in L \text{ for } i = 1 \cdots K\}$ without replacement
3. randomly select a document for each $L_i \in S$ from WIKICONTENT without replacement
4. take the top $\frac{1}{K}$ lines of the document
5. join the $K$ sections into a single document.

Figure 6.1: Process for generating documents for the WIKIPEDIAMULTI dataset.

The dumps we used are from July – August 2010.

To generate WIKIPEDIAMULTI, we first normalized the raw mediawiki documents. Mediawiki documents typically contain one paragraph per line, interspersed with structural elements. We filtered each document to remove all structural elements, and only kept documents that exceeded 2500 bytes after normalization. This yielded a collection of around 500,000 documents in 156 languages. From this initial document set (hereafter referred to as WIKICONTENT), we only retained languages that had more than 1000 documents (44 languages), and generated documents for WIKIPEDIAMULTI using the process in Figure 6.1.

As a result of the procedure, the relative proportion of each language in a multilingual document tends not to be uniform, as it is conditioned on the length of the original document from which it was sourced, independent of the other $K - 1$ for the

| System | $\mathcal{P}_M$ | $\mathcal{R}_M$ | $\mathcal{F}_M$ | $\mathcal{P}_\mu$ | $\mathcal{R}_\mu$ | $\mathcal{F}_\mu$ |
|---|---|---|---|---|---|---|
| `SegLang` | .809 | .975 | .875 | .771 | .975 | .861 |
| `Linguini` | .853 | .772 | .802 | .838 | .774 | .805 |
| Our method | .962 | .954 | .957 | .963 | .955 | .959 |

Table 6.3: Results on the WikipediaMulti dataset.

other languages that it was combined with. Overall, the average document length is 5500 bytes (standard deviation = 3800 bytes). Due to rounding up in taking the top $\frac{1}{k}$ lines (step 4), documents with higher $K$ tend to be longer (6200 bytes for $K = 5$ vs 5100 bytes for $K = 1$).

The WikipediaMulti dataset contains training, development and test partitions. The training partition consists of 5000 monolingual (i.e. $K = 1$) documents. The development partition consists of 5000 documents, 1000 documents for each value of $K$ where $1 \leq K \leq 5$. The test partition contains 200 documents for each $K$, for a total of 1000 documents. There is no data overlap between any of the partitions.

### 6.4.1 Results over WikipediaMulti

We trained each system using the monolingual training partition, and tuned the parameters using the development partition. For `Linguini`, we tested feature counts between 10000 and 50000, and found that the effect was relatively small. We thus use 10000 features as the optimum value. For `SegLang`, we tested values for threshold $t$ between 0.01 and 0.20. Increasing the threshold increases precision at the expense of recall. We found that the maximal macro-averaged F-score is attained when $t = 0.06$. Finally, for our method we tested features-per-language counts between 30 and 130 and found the best performance with 120 features per language, although the actual

effect of varying this value is rather small. We tested values of the threshold $t$ for adding an extra language to $\lambda$ from 0.01 to 0.15, and found that the best results were attained when $t = 0.02$.

The results of evaluating each system on the test partition are summarized in Table 6.3. In this evaluation, our method clearly outperforms both `SegLang` and `Linguini`. The results on WikipediaMulti and ALTW2010 are difficult to compare directly due to the different compositions of the two datasets. ALTW2010 is predominantly bilingual, whereas WikipediaMulti contains documents with text in 1 – 5 languages. Furthermore, the average document in ALTW2010 is half the length of that in WikipediaMulti. Overall, we observe that `SegLang` has a tendency to over-label (despite the introduction of the $t$ parameter to reduce this effect), evidenced by high recall but lower precision. `Linguini` is inherently limited in that it is only able to detect up to 3 languages per document, causing recall to suffer on WikipediaMulti. However, it also tends to always output 3 languages, regardless of the actual number of languages in the document, hurting precision. Furthermore, even on ALTW2010 it has lower recall than the other two systems.

## 6.5 Estimating Language Proportions

In addition to detecting multiple languages within a document, our method also estimates the relative proportions of the document that are written in each language. This information may be useful for detecting documents that are candidate bitexts for training machine translation systems, since we may expect languages in the document to be present in equal proportions. It also allows us to identify the predominant

| | |
|---|---|
| **Original text** | $\texttt{the␣cat␣in␣the␣hat}$ |
| **$n$-gram features** | $\left\{\begin{array}{ll} \texttt{he␣}:2 & \texttt{the␣}:2 \\ \texttt{␣hat}:1 & \texttt{␣in␣}:1 \\ \texttt{␣th}:1 & \texttt{␣the}:1 \\ \texttt{hat}:1 & \texttt{he␣c}:1 \\ \texttt{in␣t}:1 & \texttt{n␣th}:1 \end{array}\right\}$ |
| **Emission rate** | $\frac{\#\text{bytes}}{\#\text{tokens}} = \frac{18}{12} = 1.5$ bytes/token |

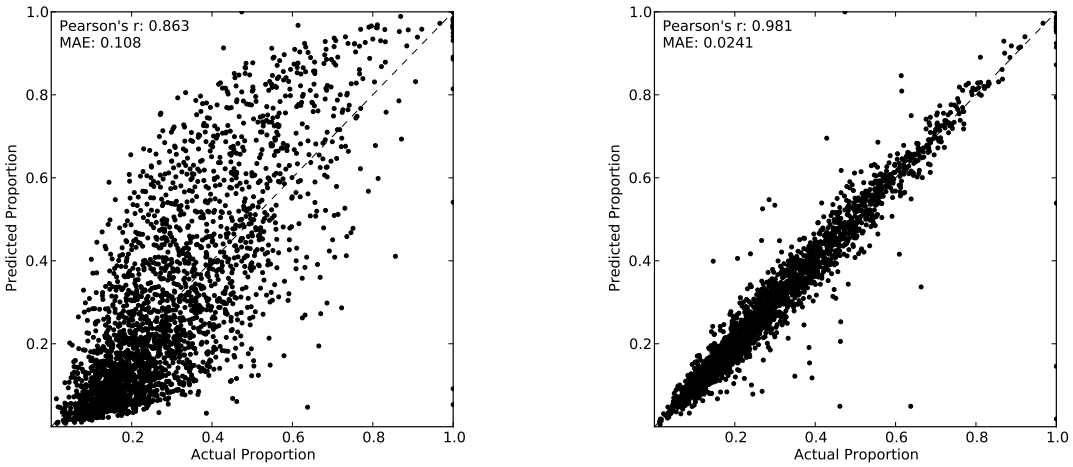
Figure 6.2: Example of calculating $n$-gram emission rate for a text string.

language of a document.

A core element of our model of a document is a distribution over a set of labels. Since each label corresponds to a language, as a first approximation, we take the probability mass associated with each label as a direct estimate of the proportion of the document written in that language. We examine the results for predicting the language proportions in the test partition of WikipediaMulti. Mapping label distributions directly to language proportions produces excellent results, with a Pearson's $r$ value of 0.863 and an MAE of 0.108.

Although labels have a one-to-one correspondence with languages, the label distribution does not actually correspond directly to the language proportion, because the distribution estimates the proportion of byte $n$-gram sequences associated with a label and not the proportion of bytes directly. The same number of bytes in different languages can produce different numbers of $n$-gram sequences, because after feature selection not all $n$-gram sequences are retained in the feature set. Hereafter, we refer to each $n$-gram sequence as a *token*, and the average number of tokens produced per byte of text as the *token emission rate*.

We estimate the per-language token emission rate (Figure 6.2) using the training




(a) without emission rate correction

(b) with emission rate correction

Figure 6.3: Scatterplot of the predicted vs. actual language proportions in a document for the test partition of WikipediaMulti (predictions are from our method; each point corresponds to a document-language pair).

partition of WikipediaMulti. To improve our estimate of the language proportions, we correct our label distribution using estimates of the per-language token emission rate $R_{L_i}$ in bytes per token for $L_i \in L$. Assume that a document $D$ of length $|D|$ is estimated to contain $K$ languages in proportions $P_i$ for $i = 1 \cdots K$. The corrected estimate for the proportion of $L_i$ is:

$$Prop(L_i) = \frac{P_i \times R_{L_i}}{\sum_{j=1}^{K} (P_j \times R_{L_j})} \tag{6.8}$$

Note that the $|D|$ term is common to the numerator and denominator and has thus been eliminated.

This correction improves our estimates of language proportions. After correction, the Pearson's $r$ rises to 0.981, and the MAE is reduced to 0.024. The improvement is

most noticeable for language – document pairs where the proportion of the document in the given language is about 0.5, as can be seen in Figure 6.3.

## 6.6 Detecting Real-world Multilingual Documents

So far, we have demonstrated the effectiveness of our proposed approach using synthetic data. The results have been excellent, and in this section we validate the approach by applying it to a real-world task that has recently been discussed in the literature. Yamaguchi and Tanaka-Ishii (2012) and King and Abney (2013) both observe that in trying to gather linguistic data for "non-major" languages from the web, one challenge faced is that documents retrieved often contain sections in another language. `SegLang` (the solution of Yamaguchi and Tanaka-Ishii (2012)) concurrently detects multilingual documents and segments them by language, but the approach is computationally expensive and has a tendency to over-label (Section 6.4). On the other hand, the solution of King and Abney (2013) is incomplete, and they specifically mention the need for an automatic method "to examine a multilingual document, and with high accuracy, list the languages that are present in the document". In this section, we show that our method is able to fill this need. We make use of manually-annotated data kindly provided to us by Ben King, which consists of 149 documents containing 42 languages retrieved from the web using a set of targeted queries for low-density languages. Note that the dataset described in King and Abney (2013) was based on manual confirmation of the presence of English in addition to the low-density language of primary interest; our dataset contains these bilingual documents as well as monolingual documents in the low-density language of interest. Our purpose in

| **System** | $\mathcal{P}$ | $\mathcal{R}$ | $\mathcal{F}$ |
|---|---|---|---|
| Baseline | 0.719 | 1.000 | 0.837 |
| `SegLang` | 0.779 | 0.991 | 0.872 |
| `Linguini` | 0.729 | 0.981 | 0.837 |
| Our method | 0.907 | 0.916 | 0.912 |

Table 6.4: Detection accuracy for English-language inclusion in web documents from targeted web crawls for low-density languages.

this section is to investigate the ability of automatic systems to select this subset of bilingual documents. Specifically, given a collection of documents retrieved for a target language, the task is to identify the documents that contain text in English in addition to the target language. Thus, we re-train each system for each target language, using only training data for English and the target language. We reserve the data provided by Ben King for evaluation, and train our methods using data separately obtained from the Universal Declaration of Human Rights (UDHR). Where UDHR translations for a particular language were not available, we used data from Wikipedia or from a bible translation. Approximately 20 – 80 kB of data were used for each language. As we do not have suitable development data, we made use of the best parameters for each system from the experiments on WikipediaMulti.

We find that all 3 systems are able to detect that each document contains the target language with 100% accuracy. However, systems vary in their ability to detect if a document also contains English in addition to the target language. The detection accuracy for English-language inclusion is summarized in Table 6.4.[6] For compar-

[6]Note that Table 6.2 and Table 6.3 both report macro and micro-averaged results across a number of languages. In contrast Table 6.4 only reports results for English, and the values are not directly comparable to our earlier evaluation.

ison, we include a heuristic baseline based on labeling all documents as containing English. We find that, like the heuristic baseline, `SegLang` and `Linguini` both tend to over-label documents, producing false positive labels of English, resulting in increased recall at the expense of precision. Our method produces less false positives (but slightly more false negatives), and thus attains the best $\mathcal{F}$ for detecting English inclusions. Error analysis suggests that the false negatives for our method generally occur where a relatively small proportion of the document is written in English.

### 6.6.1 Cross-domain Training Data

In Chapter 4, we investigated the effect of variation within a language between different sources of text on the accuracy of a language identifier applied to a different text source than the one on which it was trained. We found that existing identifiers failed to generalize across text sources. In Chapter 5, we investigated the underlying reasons for this and developed a document representation that takes into account both the language and the source of training documents, and showed that this representation is more robust to the variation in a language across text sources when combined with a variety of learning algorithms. In this chapter, we have addressed another aspect of generalized LangID, and developed a method to identify documents that contain text in more than one language, the languages present, and the relative proportion of each. In the early part of this chapter, we demonstrated the method using synthetic training data, generated using documents from Wikipedia. Rather than generate synthetic multilingual documents in multiple domains, we have instead provided a further evaluation using real-world multilingual documents. The evalua-

| **Train Data** | **Repr** | $\mathcal{P}_M$ | $\mathcal{R}_M$ | $\mathcal{F}_M$ | $\mathcal{P}_{\texttt{en}}$ | $\mathcal{R}_{\texttt{en}}$ | $\mathcal{F}_{\texttt{en}}$ |
|---|---|---|---|---|---|---|---|
| UDHR | $\mathbb{IG}_{lang}$ | 0.998 | 0.998 | 0.998 | 0.907 | 0.916 | 0.912 |
| UDHR + ODIN | $\mathbb{IG}_{lang}$ | 0.831 | 0.776 | 0.792 | 0.907 | 0.916 | 0.912 |
| UDHR + ODIN | $\mathbb{LD}$ | 0.998 | 0.999 | 0.998 | 0.910 | 0.944 | 0.927 |

Table 6.5: LangID accuracy on multilingual web documents from targeted web crawls for low-density languages. The UDHR result is replicated from Table 6.4 for comparison.

tion presented in Section 6.6 uses training data drawn primarily from UDHR (with some documents from Wikipedia and bible translations where no UDHR data was available for a particular language). The target data is from an entirely different source: it consists of real-world multilingual web pages. One challenge in demonstrating the utility of the cross-domain feature selection we developed in Section 5.4 is the lack of training data from other sources. The work of King and Abney (2013) is specifically targeted at collecting data for under-resourced languages, and as such it is unsurprising that training data for such languages is limited. However, there have been other projects that have aimed to collect text in under-resourced languages, and an example we discussed in Section 2.5.2 is ODIN, the Online Database of INterlinear text (Xia *et al.* 2010a). We cross-referenced the languages covered by ODIN and by our evaluation dataset, and found that data for 24 of the 42 languages was available in ODIN. Thus, in this section, we report on the results of adding the training data from ODIN to our existing training data. We again note that our existing training data is primarily from UDHR, with a small number of documents from Wikipedia and bible translations where no UDHR data was available. For brevity, we refer to this initial training set as simply UDHR data.

Table 6.5 shows the results of adding the additional training data from ODIN to our existing UDHR training data. We present both the macro-averaged precision, recall and F-score ($\mathcal{P}_M$,$\mathcal{R}_M$,$\mathcal{F}_M$), as well as the values for English only, noting that in the UDHR result replicated from Table 6.4 the detection accuracy for all languages other than English was 100%. We report the results for both $\mathbb{IG}_{lang}$ and $\mathbb{LD}$ feature selection to emphasize the point that the improved accuracy is not simply due to an increase in the quantity of training data. For the experiments in Section 6.6, the training data comes from a single source and thus the feature selection is based only on information gain with respect to language ($\mathbb{IG}_{lang}$). Simply adding the ODIN training data to the pool of training data and only selecting features on the basis of information gain with respect to language results in a classifier that performs substantially worse than without the ODIN data. The decrease in performance is actually due to languages other than English – the accuracy on English is unchanged between UDHR and UDHR + ODIN, but the macro-averaged precision, recall and F-score are reduced due to errors made in detecting the other languages present. Feature selection through $\mathbb{LD}$ has exactly the desired effect, it selects a feature set that is representative of each language across the multiple sources of text. Interestingly, the results on English detection are improved by the inclusion of the ODIN training data, despite ODIN not containing any training data for English. Furthermore, ODIN only adds training data for a subset of the languages in the test data, and hence it stands to reason that a further improvement in performance could be expected if training data for the remaining languages from another domain were also available.

## 6.7 Chapter Summary

We have presented a system for LangID in multilingual documents using a generative mixture model inspired by supervised topic modeling algorithms, combined with a document representation based on previous research in LangID for monolingual documents. We showed that the system outperforms alternative approaches from the literature on synthetic data, as well as on real-world data from related research on linguistic corpus creation for low-density languages using the web as a resource. We also showed that our system is able to accurately estimate the proportion of the document written in each of the languages identified. Finally, we showed that we are able to improve accuracy of LangID on real-world multilingual documents by integrating our approach to cross-domain feature selection that we developed in Chapter 5.

We have made a full reference implementation of our system freely available,[7] as well as the synthetic dataset prepared for this chapter (Section 6.4), in order to facilitate the adoption of this technology and further research in this area.

[7] `https://github.com/saffsd/polyglot`

# Chapter 7

# Twitter: A Case Study in "Off-the-Shelf" LangID

In Chapter 2, we saw how work to date has focused on various aspects of the overall task of LangID, and has generally offered very promising results, to the extent that some have claimed LangID is a solved task (McNamee 2005). A large number of different methods for LangID have been proposed, and a number of implementations of these methods are available as general-purpose "off-the-shelf" LangID systems (Section 2.4). In this chapter, we present a case study in applying such "off-the-shelf" LangID to Twitter,[1] a popular microblogging service. Twitter has

This chapter is based on work previously published as:

Lui, Marco, and Timothy Baldwin. 2014. Accurate Language Identification of Twitter Messages. In *Proceedings of the 5th Workshop on Language Analysis in Social Media*, 17 – 25, Gothenburg, Sweden.

[1] `http://www.twitter.com`

captured the attention of various research communities as a potent data source, because of the immediacy of the information presented, the volume and variability of the data contained, the potential to analyze networking effects within the data, and the ability to (where GPS data is available) geolocate messages (Krishnamurthy *et al.* 2008). Although individual messages range from inane through mundane right up to insane, the aggregate of these messages can lead to profound insights in real-time. Examples include real-time detection of earthquakes (Sakaki *et al.* 2010), analysis of the location and prevalence of flu epidemics (Lampos *et al.* 2010; Culotta 2010), news event detection (Petrović *et al.* 2010), and diachronic prediction of election outcomes (Tumasjan *et al.* 2010).

Text analysis of social media has quickly become one of the "frontier" areas of NLP, with major conferences opening entire tracks for it in recent years. The challenges in NLP for social media are many, stemming primarily from the "noisy" nature of the content. Research indicates that English Twitter in particular is more dissimilar to the kinds of reference corpora used in NLP to date, compared to other forms of social media such as blogs and comments (Baldwin *et al.* 2013). This has led to the development of techniques to normalize Twitter messages (Han *et al.* 2013), as well as Twitter-specific approaches to conventional NLP tasks such as part-of-speech tagging (Gimpel *et al.* 2011) and information extraction (Bontcheva *et al.* 2013). Even so, a general precondition of NLP techniques is that the language of the input data is known, and this has led to interest in LangID of Twitter messages. Research has shown that "off-the-shelf" LangID systems appear to perform fairly well on Twitter (Lui and Baldwin 2012), but Twitter-specific systems seem to perform better (Carter

*et al.* 2013; Tromp and Pechenizkiy 2011; Bergsma *et al.* 2012; Goldszmidt *et al.* 2013).

Twitter recognizes the utility of language metadata in enabling new applications, and as of March 2013 includes language predictions with results from its API (Roomann-Kurrik 2013). These predictions are not perfect (see Section 7.2.2), and at time of writing do not cover some languages (e.g. Romanian). Furthermore, some research groups have collected a substantial cache of Twitter data from before the availability of built-in predictions. Motivated by the need to work with monolingual subsets of historical data, we investigate the most practical means of carrying out LangID of Twitter messages, balancing accuracy with ease of implementation. In this work, we present an evaluation of "off-the-shelf" language identifiers, combined with techniques that have been proposed for boosting accuracy on Twitter messages.

A major challenge that we have had to overcome is the lack of annotated data for evaluation. Bergsma *et al.* (2012) point out that in LangID research on microblog messages to date, only a small number of European languages has been considered. Baldwin and Lui (2010a) showed that, when considering full documents, good performance on just European languages does not necessarily imply equally good performance when a larger set of languages is considered. This does not detract from work to date on European languages (Tromp and Pechenizkiy 2011; Carter *et al.* 2013), but rather highlights the need for further research in LangID for microblog messages.

Manual annotation of Twitter messages is a challenging and laborious process. Furthermore, Twitter is highly multilingual, making it very difficult to obtain annota-

tors for all of the languages represented. Previous work has attempted to crowdsource part of this process (Bergsma *et al.* 2012), but such an approach requires substantial monetary investment, as well as care in ensuring the quality of the final annotations. In this chapter, we propose an alternative, "mostly-automated" approach to gathering language-labeled Twitter messages for evaluating LangID. A corpus constructed by direct application of automatic LangID to Twitter messages would obviously be unsuitable for evaluating the accuracy of LangID tools. Even with manual post-filtering, the remaining dataset would be biased towards messages that are easy for automated systems to classify correctly. The novelty of our approach is to leverage user identity, allowing us to construct a corpus of language-labeled Twitter messages without using automated tools to determine the languages of the messages. This quality makes the corpus suitable for use in the evaluation of automated LangID of Twitter messages.

The main contributions of this chapter are: (1) we provide a manually-labeled dataset of Twitter messages, adding Chinese (zh) and Japanese (ja) to the set of Twitter messages with human annotation for language; (2) we provide a second dataset constructed using a mostly-automated approach, covering 65 languages; (3) we detail the method for constructing the dataset; (4) we provide a comprehensive empirical evaluation of the accuracy of off-the-shelf LangID systems on Twitter messages, using published datasets in addition to the new datasets we have introduced; and (5) we discuss and evaluate a simple voting-based ensemble for LangID, and find that it outperforms any individual system to achieve state-of-the-art results.

| Dataset | Reference | Message Count | Languages (ISO639-1) |
|---|---|---|---|
| T-Benelearn | Tromp and Pechenizkiy (2011) | 9066 | de es en fr it nl (6) |
| T-SCarter | Carter *et al.* (2013) | 5000 | de es en fr nl (5) |
| Bergsma | Bergsma *et al.* (2012) | 13190 | ar bg fa hi mr ne ru uk ur (9) |
| T-ZhEnJa | new | 3016 | en ja zh (3) |

Table 7.1: Datasets of Language-labeled Twitter messages.

## 7.1 Datasets

In Chapter 2, we discussed some work to date on LangID on Twitter data. Some authors have released accompanying datasets; the dataset used by Tromp and Pechenizkiy (2011) was made available in its entirety, consisting of 9066 messages in 6 Western European languages. Other authors have released message identifiers with associated language labels, including Carter *et al.* (2013), with 5000 identifiers in 5 Western European languages, and Bergsma *et al.* (2012), providing 13190 identifiers across 9 languages from 3 language families (Arabic, Cyrillic, Devanagari). To date, only the dataset of Tromp and Pechenizkiy (2011) has been used by other researchers (Goldszmidt *et al.* 2013). With the kind co-operation of the authors, we have obtained the full datasets of Carter *et al.* (2013) and Bergsma *et al.* (2012), allowing us to present the most extensive empirical evaluation of LangID of Twitter messages to date. However, the total set of languages covered is still very small. In Section 7.1.1, we present our own manually-annotated dataset, adding Chinese (zh) and Japanese (ja) to the languages that have manually-annotated data. We discuss some of the challenges that manual annotation poses, and in Section 7.1.2 we introduce a technique for evaluation dataset construction that helps us tackle these challenges.

| | English | Chinese | Japanese |
|---|---|---|---|
| Initial | 0.906 | 0.773 | 0.989 |
| Post-review | 0.930 | 0.916 | 0.998 |

Table 7.2: Fleiss' kappa over annotations for TWITTER.

### 7.1.1 Manual Annotation of T-ZhEnJa

A manual approach to constructing a LangID dataset from Twitter data is difficult due to the wide variety of languages present on Twitter – Bergsma *et al.* (2012) report observing 65 languages in a 10M message sample, and Baldwin *et al.* (2013) report observing 97 languages in a 1M message sample. While this is encouraging in terms of sourcing data for lower-density languages, the distribution of languages is Zipfian, and the relative proportion of data in most languages is very small. Manually retrieving all available messages in a language would require a native speaker to view and reject a huge number of messages in other languages in order to collect the small number that are written in the target language. We initially attempted this, building T-ZHENJA, a dataset derived from a set of 5000 messages randomly sampled from a larger body of 622192 messages collected from the Twitter streaming API over a single 24-hour period in August 2010. The messages are a 1% representative sample of the total public messages posted on that day. Each of the 5000 selected messages was annotated by speakers of three languages, English, Japanese and Mandarin Chinese. For each message, three annotators were asked if the message contained any text in languages which they spoke, as well as if it appeared to contain text in (unspecified) languages which they did not speak. The latter label was introduced in order to make a distinction between text in languages not spoken by our annotators

(e.g. Portuguese) and text with no linguistic content (e.g. URLs). After the initial annotation, annotators were asked to review messages where there was disagreement, and messages were assigned labels given by a majority of annotators post-review. Inter-annotator agreement (Table 7.2) is strong for the task: only 20 out of 5000 messages have less than 80% majority in annotations. In many instances, the disagreement was due to messages consisting entirely of a short sequence of hanzi/kanji, which both Chinese and Japanese speakers recognized as valid (these messages are excluded from our set of labeled messages). Out of the 5000 messages, 1953 (39.1%) were labeled as English, 16 were labeled as Chinese (0.3%) and 1047 were labeled as Japanese (20.9%), for a total of 3016 labeled messages.

A total of 8 annotators each invested 2 – 4 hours in this annotation task, and the final dataset only covers 3 languages (which includes the top-2 highest-density languages in Twitter). Obviously, constructing a dataset of language-labeled Twitter messages is a labor-intensive process, and the lower density the language, the more expensive our methodology becomes (as more and more documents need to be looked over to find documents in the language of interest). Ideally, we would like to be able to use some form of automated LangID to accelerate the process without biasing the data towards easy-to-classify messages.

### 7.1.2 A Broad-Coverage Twitter Corpus

Based on our discussion so far, our desiderata for a LangID dataset of Twitter messages are as follows: (1) achieve broader coverage of languages than existing datasets; (2) minimize manual annotation; and (3) avoid bias induced by selecting

**Algorithm 2** Procedure for building a Twitter LangID dataset.

$U \leftarrow$ active users
$L^{accept}, M^{accept}, U^{accept} \leftarrow \{\}, \{\}, \{\}$
**for each** $u \in U$ **do**
  $M_u \leftarrow$ all messages by user $u$
  $M_u^{main}, M_u^{heldout} \leftarrow \text{RandomSplit}(M_u)$
  $L_u \leftarrow \{\}$
  **for each** $m \in M_u^{main}$ **do**
    $m_l \leftarrow \text{LangID}(m)$
    **if** $m_l \neq$ unknown **then**
      $L_u \leftarrow L_u \cup \{m_l\}$
    **end if**
  **end for**
  **if** $len(L_u) = 1$ **then**
    $U^{accept} \leftarrow U^{accept} \cup \{(u, L_u)\}$
    $L^{accept} \leftarrow L^{accept} \cup L_u$
  **end if**
**end for**
**for each** $l \in L_{accept}$ **do**
  $U^{sample} \leftarrow \text{Sample}(U_l^{accept}, K)$
  **for each** $u \in U^{sample}$ **do**
    $M^{sample} \leftarrow \text{Sample}(M_u^{heldout}, N)$
    $M^{accept} \leftarrow M^{accept} \cup \{(M^{sample}, l)\}$
  **end for**
**end for**
**return** $M^{accept}$

messages using LangID. (2) and (3) may seem to be conflicting objectives, but we sidestep the problem by first identifying monolingual users, then produce a dataset by sampling messages by these users from a held-out collection.

The overall workflow for constructing a dataset is summarized in Algorithm 2. For each user we consider, we divide all their messages into two disjoint sets. One set ($M_u^{main}$) is used to determine the language(s) spoken by the user. If only one language is detected, the user is added to a pool of candidate users ($U^{accept}$). A fixed number of users is sampled for each language ($U^{sample}$), and for each sampled user a

fixed number of messages is sampled from the held-out set ($M_u^{heldout}$) and added to the final dataset. We sample a fixed number of users per language to limit the amount of data in the more-frequent languages, and we only sample a small number of messages per user in order to avoid biasing the dataset towards the linguistic idiosyncrasies of any specific individual. For both sampling steps, if the number of items available is less than the number required, all the available items are returned.

Algorithm 2 uses automated LangID to detect the language of messages in $M_u^{main}$ (line 8). The accuracy of this identifier is not critical, as any misclassifications for a monolingual user would cause them to be rejected, as they would appear multilingual. Hence, the risk of false positives at the user-level LangID is very low. However, incorrectly rejecting users reduces the pool of data available for sampling, so a higher-accuracy solution is preferable. We compared the performance of 8 off-the-shelf (i.e. pre-trained) LangID systems to determine which would be the most suitable for this role. These systems are described in more detail in Section 2.4, and in the following overview we only provide a quick recap of the characteristics of each.

**`langid.py`** (Lui and Baldwin 2012): an $n$-gram feature set selected using data from multiple sources, combined with a multinomial naive Bayes classifier. `langid.py` is based on early work that was adapted into Chapter 4 and Chapter 5, and is an implementation of the document representation developed in Section 5.4 combined with a naive Bayes classifier.

**`ChromeCLD`** (Sites 2013b): the language identifier embedded in the Chrome web browser;[2] it uses a naive Bayes classifier and script-specific tokenization strate-

[2] `http://www.google.com/chrome`

gies.

**LangDetect** (Nakatani 2010b): a naive Bayes classifier, using a character $n$-gram based representation without feature selection, with a set of normalization heuristics to improve accuracy.

**LDIG** (Nakatani 2012): a Twitter-specific LangID tool, which uses a document representation based on tries, combined with normalization heuristics and Bayesian classification, trained on Twitter data.

**whatlang** (Brown 2013): a vector-space model with per-feature weighting over character $n$-grams.

**YALI** (Majliš 2012): computes a per-language score using the relative frequency of a set of byte $n$-grams selected by term frequency.

**TextCat** (Scheelen 2003): an implementation of Cavnar and Trenkle (1994), which uses an ad-hoc rank-order statistic over character $n$-grams.

**MSR-LID** (Goldszmidt *et al.* 2013): based on rank-order statistics over character $n$-grams, and Spearman's $\rho$ to measure correlation. Twitter-specific training data is acquired through a bootstrapping approach. We use the 49-language model provided by the authors, and the best parameters reported in the paper.

We investigated the performance of the systems using manually-labeled datasets of Twitter messages (Table 7.3), including the T-ZhEnJa set described in Section 7.1.1.[3]

[3]We do not limit the comparison to languages supported by each system as this would bias evaluation towards systems that support few languages that are easy to discriminate.

| Dataset | `langid.py` | `ChromeCLD` | `LangDetect` | `LDIG` | `whatlang` | `YALI` | `TextCat` | `MSR-LID` |
|---|---|---|---|---|---|---|---|---|
| T-Benelearn | 0.983 | 0.972 | 0.959 | 0.986 | 0.950 | 0.911 | 0.814 | 0.983 |
| T-SCarter | 0.917 | 0.902 | 0.891 | 0.943 | 0.834 | 0.824 | 0.510 | 0.927 |
| Bergsma | 0.847 | 0.911 | 0.923 | *0.000* | 0.719 | *0.428* | *0.046* | *0.546* |
| T-ZhEnJa | 0.871 | 0.884 | 0.831 | *0.315* | 0.622 | 0.877 | 0.313 | 0.848 |

Table 7.3: Macro-averaged F-score on manually-annotated Twitter datasets. *Italics* denotes results where the dataset contains languages not supported by the identifier.

We find that all the systems tested perform well on T-Benelearn, with the exception of `TextCat`. T-SCarter covers a very similar set of languages to T-Benelearn, yet all systems consistently perform worse on it. This suggests that T-Benelearn is biased towards messages that LangID systems are likely to identify correctly (also observed by Goldszmidt *et al.* (2013)). This is due in part to the post-processing applied to the messages, but also suggests a bias in how messages were selected. `LDIG` is the best performer on T-Benelearn and T-SCarter, albeit falling slightly short of the 99.1% accuracy reported by the author (Nakatani 2012). However, it is only trained on 17 languages and thus is not able to fully support Bergsma and T-ZhEnJa, and so we cannot draw any conclusions on whether the method will generalize well to more languages. The system that supports the most languages by far is `whatlang`, but as a result its accuracy on Twitter messages suffers. Manual analysis suggests this is due to Twitter-specific "noise" tipping the model in favor of lower-density languages. On Bergsma, `LangDetect` is the best performer, likely due to its specific heuristics for distinguishing certain language pairs (Nakatani 2010b), which happen to be present in the Bergsma dataset. Overall, in their off-the-shelf configuration, only three systems (`langid.py`, `ChromeCLD`, `LangDetect`) perform consistently well on LangID of Twitter messages. Even so, the macro-averaged F-Scores observed were

| Dataset | Single Best | | Voting | | 3-System |
|---|---|---|---|---|---|
| | System | F-score | Systems | F-score | F-score |
| T-Benelearn | `LDIG` | 0.986 | `ChromeCLD`, `MSR-LID`, `LDIG` | 0.992 | 0.986 |
| T-SCarter | `LDIG` | 0.943 | `MSR-LID`, `langid.py`, `LDIG` | 0.948 | 0.927 |
| Bergsma | `LangDetect` | 0.923 | `ChromeCLD`, `LangDetect`, `langid.py` | 0.935 | 0.935 |
| T-ZhEnJa | `ChromeCLD` | 0.884 | `ChromeCLD`, `MSR-LID`, `LDIG`, `YALI`, `langid.py` | 0.969 | 0.941 |

Table 7.4: System combination by majority voting. All combinations of 3, 5 and 7 systems were considered. For each dataset, we report the single-best system, the best combination, and F-score of the majority-vote combination of `langid.py`, `ChromeCLD` and `LangDetect`.

as low as 83%, indicating that whilst performance is good, the problem of LangID of Twitter messages is far from solved.

Given that the set of languages covered and accuracy varies between systems, we investigated a simple voting-based approach to combining the predictions. For each dataset, we considered all combinations of 3, 5, and 7 systems, combining the predictions using a simple majority vote. The single-best combination for each dataset is reported in Table 7.4. In all cases, the macro-averaged F-score is improved upon, showing the effectiveness of the voting approach. Hence, for purposes of LangID in Algorithm 2, we chose to use a majority-vote ensemble of `langid.py`, `ChromeCLD` and `LangDetect`, a combination that generally performs well on all datasets.[4] Where all 3 systems disagree, the message is labeled as unknown. In determining whether a user is multilingual, messages labeled unknown are discarded rather than being treated as a distinct language, as disagreement is usually a sign that the message contains specific features that confuse a particular classifier. Rejecting unknown

[4] `MSR-LID` was excluded due to technical difficulties in applying it to a large collection of messages because of its oversized model.

messages thus reduces the incidence of wrongly rejecting monolingual users due to a specific misclassifying a particular message without causing any multilingual users to be detected as monolingual, as multilingual users generally have at least one message in each language that they use. This voting-based ensemble of classifiers is hereafter referred to as VOTING.

To build our final dataset, we collected all messages by active users from the 1% feed made available by Twitter over the course of 31 days, between 8 January 2012 and 7 February 2012. We deemed users active if they had posted at least 5 messages in a single day on at least 7 different days in the 31-day period we collected data for. This gave us a set of approximately 2M users. For each user, we partitioned their messages (RandomSplit in Algorithm 2) by selecting one day at random. All of the messages posted by the user on this day were treated as heldout data ($M_u^{heldout}$), and the remainder of the user's messages ($M_u^{main}$) were used to determine the language(s) spoken by the user. The day chosen was randomly selected per-user to avoid any bias that may be introduced by messages from a particular day or date. Of the active users, we identified 85.0% to be monolingual, covering a set of 65 languages. 50.6% of these users spoke English (en), 14.1% spoke Japanese (ja), and 13.0% spoke Portuguese (pt); this user-level language distribution largely mirrors the message-level language distribution reported by Baldwin *et al.* (2013) and others. From this set of users, we randomly selected up to 100 users per language, leaving us with a pool of 26011 held-out messages from 2914 users. Manual inspection of these messages revealed a number of English messages mislabeled with another language, indicating that even predominantly monolingual users occasionally introduce English into their

| Tool | Without Cleaning | | | | With Cleaning | | | |
|---|---|---|---|---|---|---|---|---|
| | P | R | F | Acc | P | R | F | Acc |
| `langid.py` | 0.767 | 0.861 | 0.770 | 0.842 | 0.759 | 0.861 | 0.766 | 0.840 |
| `ChromeCLD` | 0.852 | 0.814 | 0.806 | 0.775 | 0.866 | 0.823 | 0.820 | 0.780 |
| `LangDetect` | 0.618 | 0.680 | 0.626 | 0.839 | 0.623 | 0.687 | 0.634 | 0.854 |
| `LDIG` | 0.167 | 0.239 | 0.189 | 0.447 | 0.167 | 0.239 | 0.189 | 0.447 |
| `whatlang` | 0.749 | 0.655 | 0.663 | 0.624 | 0.739 | 0.667 | 0.663 | 0.623 |
| `YALI` | 0.441 | 0.564 | 0.438 | 0.710 | 0.449 | 0.560 | 0.443 | 0.705 |
| `TextCat` | 0.327 | 0.245 | 0.197 | 0.257 | 0.316 | 0.295 | 0.230 | 0.316 |
| `MSR-LID` | 0.533 | 0.609 | 0.536 | 0.848 | 0.533 | 0.609 | 0.536 | 0.848 |
| VOTING | 0.920 | 0.876 | 0.887 | 0.861 | 0.919 | 0.883 | 0.889 | 0.868 |

Table 7.5: Macro-averaged Precision/Recall/F-score, as well as message-level accuracy for each system on TWITUSER. The right side of the table reports results after applying message-level cleaning (Tromp and Pechenizkiy 2011).

online communications. Such messages are generally entirely English, with code-switching (i.e. multiple languages in the same message) very rarely observed. In order to eliminate mislabeled messages, we applied all 8 systems to this pool of 26011 messages. Where at least 5 systems agree and the predicted language does not match the user's language, we discarded the message. Where 3 or 4 systems agree, we manually inspected the messages and eliminated those that were clearly mislabeled (this is the only manual step in the construction of this dataset). Overall, we retained 24220 messages (93.1%). From these, we sampled up to 5 messages per unique user, producing a final dataset of 14178 messages across 65 languages (hereafter referred to as the TWITUSER dataset).

## 7.2 Evaluating Off-the-Shelf LangID

Given TwitUser, our broad-coverage Twitter corpus, we return to the task of examining the performance of the off-the-shelf LangID systems we discussed in Section 7.1.2 (Table 7.5, left side). In terms of macro-averaged F-score across the full set of 65 languages, `ChromeCLD` is the single best-performing system. Unlike `langid.py` and `LangDetect`, `ChromeCLD` does not always produce a prediction, and instead has an in-built threshold for it to output a prediction of "unknown". This is reflected in the elevated precision, at the expense of decreased recall and message-level accuracy. Systems like `langid.py` which always make a prediction have reduced precision, balanced by increased recall and message-level accuracy. As with the manually-annotated datasets, we experimented with a simple voting-based approach to combining multiple classifiers. We again experimented with all possible combinations of 3, 5 and 7 classifiers, and found that on TwitUser, a majority-vote ensemble of `ChromeCLD`, `langid.py` and `LangDetect` attains the best macro-averaged F-score, and also outperforms any individual system on all of the metrics considered. We note that this is exactly the Voting ensemble of Section 7.1.2, validating its choice as LangID($m$) in Algorithm 2.

### 7.2.1 Adapting Off-the-Shelf LangID to Twitter

Tromp and Pechenizkiy (2011) propose to remove links, usernames, hashtags and smilies before attempting LangID, as they are Twitter specific. We experimented with applying this cleaning procedure to each message body before passing it to our off-the-shelf systems (Table 7.5, right side). For `LDIG` and `MSR-LID`, the results

are exactly the same with and without cleaning. These two systems are specifically targeted at Twitter messages, and thus may include a similar normalization as part of their processing pipeline. This also suggests that the systems do not leverage this Twitter-specific content in making predictions. Other systems generally show a small improvement with cleaning, except for `langid.py`. The VOTING ensemble also benefits from cleaning, due to the improvement in two of its component classifiers (`ChromeCLD` and `LangDetect`). This cleaning procedure is trivial to implement, so despite the improvement being small, it may be worth implementing if adapting off-the-shelf language identifiers to Twitter messages.

Goldszmidt *et al.* (2013) suggest bootstrapping a Twitter-specific language identifier using an off-the-shelf language identifier and an unlabeled collection of Twitter messages. We tested this approach, using the 3 systems that provide tools to generate new models from labeled data (`LangDetect`, `langid.py` and `TextCat`). We constructed bootstrap collections by: (1) using the off-the-shelf tools to directly identify the language of messages; and (2) using Algorithm 2. Overall, the bootstrapped identifiers are not better than their off-the-shelf counterparts. For `TextCat` there is an increase in accuracy using bootstrapped models, but the accuracy of `TextCat` with bootstrapped models is still inferior to `LangDetect` and `langid.py` in their off-the-shelf configuration. For `LangDetect`, utilizing bootstrapped models does not always increase the accuracy of LangID of Twitter messages. Where it does help, the bootstrap collections that are effective vary with the target dataset. For `langid.py`, none of the bootstrapped models outperformed the off-the-shelf model. This suggests that for LangID, the same features that are predictive of language in other domains are

| Dataset | Period | Proportion |
|---|---|---|
| T-SCarter | Jan – Apr 2010 | 76.4% |
| Bergsma | May 2007 – Feb 2012 | 92.2% |
| TwitUser | Jan – Feb 2012 | 79.7% |

Table 7.6: Proportion of messages from each dataset that were still accessible as of August 2013.

equally applicable to Twitter messages, and that the cross-domain feature selection procedure we developed in Section 5.4 that is implemented by `langid.py` (Lui and Baldwin 2011) is able to identify these features effectively.

Bontcheva *et al.* (2013) report positive results from the integration of LangID priors (Carter *et al.* 2013), but we did not experiment with them, as the calculation of priors is relatively expensive compared to the other adaptations we have considered, in terms of both run time and developer effort. Furthermore, there are a number of open issues that are likely to affect the effectiveness of the priors, such as the size and the scope of the message collection used to determine the prior. This is an interesting avenue of future work but is beyond the scope of this thesis. However, we observe that priors based on user identity (e.g the "Blogger" prior) are likely to be artificially effective on TwitUser, because the messages have been sampled from users that we have identified as monolingual.

### 7.2.2 Twitter API Predictions

For T-SCarter, Bergsma and TwitUser, we have access to the original identifiers for each message, which use used to download the messages via the Twitter

API.[5] Table 7.6 reports the proportion of each dataset that is still accessible as of August 2013. For the messages that we were able to recover, the full response from the API now includes language predictions. We do not report quantitative results on the accuracy of the Twitter API predictions as the Twitter API terms of service forbid benchmarking ("You will not attempt ... to ... use or access the Twitter API ... for ... benchmarking or competitive purposes"). Furthermore, any results would be impossible to replicate: the set of messages that are accessible is likely to continue to decrease, and the accuracy of Twitter's predictions may vary as updates are made to the API.

Error analysis of the language predictions provided by the Twitter API shows that at the time of writing, for the languages supported the accuracy of the Twitter API is not substantially better than the best off-the-shelf language identifiers we examined in this paper. However, about a quarter of the languages present in TwitUser are never offered as predictions. This has implications for the precision of LangID in other languages: one notable example is poor precision in Italian, due to some Romanian messages being identified as Italian (no messages are identified as Romanian). This suggests that caution must be taken in taking the language predictions offered by the Twitter API as goldstandard. The accuracy of the predictions is not perfect, and highlights the need for further research into improving the scope and accuracy of LangID for Twitter messages.

[5] `http://dev.twitter.com`

## 7.3 Chapter Summary

In this chapter, we introduced T-ZhEnJa and TwitUser, two novel datasets of language-labeled Twitter messages. T-ZhEnJa is constructed using a conventional manual annotation approach, whereas TwitUser is constructed using a novel mostly-automated method that leverages user identity. Using these new datasets alongside three previously-published datasets, we compared 8 off-the-shelf LangID systems over Twitter messages, and found that a simple majority vote across three specific systems (`ChromeCLD`, `langid.py`, `LangDetect`) consistently outperforms any individual system. We also found that removing Twitter-specific content from messages improves the performance of off-the-shelf systems. We reported that the predictions provided by the Twitter API are not better than state-of-the-art off-the-shelf systems, and that a number of languages in use on Twitter appear to be unsupported by the Twitter API, underscoring the need for further research to broaden the scope and accuracy of LangID of Twitter messages.

# Chapter 8

# Conclusion

The central theme of this thesis, generalized LangID, deals with eliminating the assumptions that limit the applicability of LangID methods to specific settings. The problem of LangID bears many similarities to supervised text categorization, but as we discussed in Section 1.1, there are important differences that make the LangID task challenging and unique. In Chapter 1, we identified three main aspects of generalized LangID that we addressed in the course of this thesis: (1) the variation present in a language across different sources of text, the impact this has on LangID accuracy, and methods to build LangID systems robust to this variation; (2) *multilingual* documents, the reasons for which they are of interest, and methods to detect when a document contains text in more than one language, the languages present as well as the relative proportion of the document written in each language; and (3) LangID in new and challenging domains, where the documents diverge significantly from the longer, well-structured, curated text that has been used in LangID research to date.

The first issue we considered was the effect on LangID accuracy of using training

data drawn from a different source to the target domain. This is the general problem facing "off-the-shelf" LangID systems, which include pre-trained models of language. Since it is not possible to know in advance what the target text will look like, it is necessary to build language identifiers that take into account the variation of a language between sources. Our first contribution in this thesis was to demonstrate that loss of accuracy due to a mismatch between training and test data is indeed an issue for existing LangID systems. To do this, in Chapter 3 we first identified and analyzed the type of variation that exists within a single language across different sources of text. We identified linguistic and non-linguistic reasons why text in the same language might "look" different, and prepared datasets from 9 different sources covering a total of 145 languages for use in our experiments. In Chapter 4, we identified three existing systems described in the literature that could be re-trained with new training data. We gave a detailed description of each system, including some initial analysis of the similarities and differences between them.

In order to quantify the effect that language variation between text sources has on LangID accuracy, we set up a number of experiments. Our first experiment involved establishing an *in-domain* benchmark for each system on each of the text sources for which we had prepared datasets, a setting which corresponds to the typical circumstances under which LangID systems are tested. Following standard machine learning practice, we divided each dataset into training, development and test components, and tested each combination of dataset and LangID system. Our overall finding was that results were generally consistent with what we expected from previous research. We found that all systems attained high accuracy in-domain when applied to datasets

that are similar to those that have been used in previous research. However, accuracy was substantially worse in datasets that presented exhibited particular oddities, such as a high proportion of domain-specific "noise" in the form of HTML/XML markup, or extreme conditions in terms of number of languages or shortness of documents.

In our next set of experiments, we evaluated each of the three systems *cross-domain*, drawing training and test data from different sources. We investigated two variants of this approach. The first, *one-source*, selects training data from a single source and test data from a different source. The second, *all-source*, selects test data from a single source, and uses the union of all sources excluding the test source as training data. In order to meaningfully compare results from different datasets in *one-source*, we evaluated each system on the basis of the macro-averaged results across 5 specific languages that were present in all the datasets used. We found that, in comparison to the *in-domain* benchmark, *cross-domain* results were generally worse. Furthermore, there was no single combination of system and training data that produces a classifier that has good accuracy on all the target domains.

The subsequent experiment investigated the *all-source* approach to cross-domain LangID. In this setting, for each dataset held out as test data, the union of all the other datasets was used as training data. Again, the results from cross-domain LangID were substantially inferior to the results from in-domain LangID.

In summary, in Chapter 4 we demonstrated that while commonly-used LangID tools can be very effective in the *in-domain* classification setting that is common in the literature, and can also be effective *cross-domain* for certain pairings of training and test data, the general trend is that there is a substantial loss in accuracy when training

and test data come from different sources. Chapter 5 investigated the underlying reasons for this deficit and developed a strategy to mitigate the effects. We examined the three systems we used in Chapter 4 more closely, and compared them with respect to how they represented documents and languages, and the classification algorithms they implemented. We found that the representations used by the systems were actually very similar, in that they were all derived from the relative frequency of specific sets of byte sequences. The algorithms used by the three systems are different, but all three systems implement what is essentially supervised machine learning. We discussed the inductive learning hypothesis that underpins machine learning, and related it to the concept of homogeneity that is found in corpus linguistics. We introduced a method for quantifying homogeneity that is used in corpus linguistics, and adapted it to quantify the homogeneity of language across different sources of text. We found that, under a representation of text similar to that used by the three systems we investigated, the level of dissimilarity between documents in the same language from different text sources was similar to the level of dissimilarity between documents in different languages from the same text source. The heterogeneity of the same language across different text sources has serious implications for the applicability of the inductive learning hypothesis, since if language is heterogeneous across text sources then we cannot expect models learned on one text source to correctly LangID documents from a different source.

To deal with the difference in language between different sources, we drew on work in transfer learning. The cross-domain classification scenario we examined closely resembles *transductive transfer learning*, with the important difference that in transfer

learning, it is generally assumed that some unlabeled target-domain data is available at training time. Since this is not necessarily the case in cross-domain LangID, specific methods for transductive transfer learning are not applicable to cross-domain LangID. However, a common theme in transductive transfer learning is that a feature space can be divided into general and domain-specific components, and we applied this idea to develop a feature selection method that takes into account both the language of a document and the text source the document is from in order to select features that are representative of a language regardless of the text source. We showed that under such a representation, languages are much more homogeneous across different text sources. On this basis, we combined our novel representation with each of the learning algorithms used by the systems we examined, and found that in the cross-domain setting, classifiers using our novel representation substantially outperformed the existing systems when using the same training and test data. Finally, we presented an error analysis, in which we investigated a number of factors that affected the accuracy of our proposed system. We found that languages for which we had training data from at least 3 different sources were generally more accurately classifier than languages for which we had training data from less sources. We also found that the number of features selected per language in the range we investigated (50 to 300 features per language) generally had little effect on the accuracy. Finally, we found that whereas most languages performed best when using a byte 4-gram model, for specific languages a byte $n$-gram model where a mixture of sequence lengths was used was much better, due to interactions with specific encodings used.

In Chapter 6, we developed a method for LangID of multilingual documents, i.e. documents that may contain text from more than one language. The method we present is able to detect that a document is multilingual, identify the languages present and estimate the relative proportions of the document written in each language. Another key property of the method is that it is trained using only language-labeled monolingual documents, which are much more readily available than language-labeled multilingual documents. The document representation we use is based on the one we developed in Chapter 5. In Chapter 6, we initially use training data from only a single source, and hence the method from Chapter 5 is not fully applicable. However, at the end of the chapter we carry out a further experiment using training data from multiple domains which shows that the document representation developed in Chapter 5 is also beneficial in LangID of multilingual documents. The core of our multilingual language identifier is a generative mixture model which is in some ways similar to Latent Dirichlet Allocation (LDA) as applied to topic modeling. However, there are a number of differences: (1) in LDA, the tokens are words, whereas in our model the tokens are byte $n$-grams (Figure 2.1); (2) in LDA, documents are a mixture of topics, whereas in our model documents are a mixture of languages; (3) in LDA, the distribution of tokens in topics must be inferred from unlabeled data, whereas in our model the distribution of tokens in languages is estimated from labeled data; and (4) in LDA the number of topics must be specified, whereas our model infers the number of languages in a document. Similarly to LDA, we use a Gibbs sampler to recover the distribution of languages in a document. To determine the correct number of languages in a document, we introduce a greedy heuristic to find the subset

of languages that maximizes the posterior likelihood of the document. We compare our method to two existing approaches to LangID for multilingual documents, using synthetic as well as real-world datasets. Our real world dataset is based on work on corpus construction for lower-density languages using targeted web crawls, and we show that our method substantially outperforms the other two systems, which only attain near-baseline performance. We estimate the proportion of the document written in each language using the distribution over languages derived from the Gibbs sampler, adjusting the raw distribution over tokens using an estimate of the average token length per-language, and show that this substantially improves the estimate of the language proportions. Finally, using additional training data drawn from a different dataset, we show the applicability of the document representation we developed in Chapter 5 to the task of multilingual LangID.

Chapter 7 presents a case study in "off-the-shelf" LangID. In Chapter 2, we identified a range of "off-the-shelf" language identifiers, i.e. software systems that included pre-trained models, such that they could be used to predict the language of a document without the user having to provide any training data. Based on our results in Chapter 4, it is questionable whether such systems would perform well on a novel and challenging domain. We thus carried out an empirical evaluation of 8 such systems as applied to Twitter messages, which are challenging due to the short length of each message and the informal tone, which leads to many linguistic irregularities. One key challenge in evaluating LangID on Twitter is the lack of datasets of Twitter messages annotated for language that have a broad coverage of languages. We tested the systems on existing datasets of language-labeled Twitter messages and found that

performance was generally good in datasets that covered a small number of western European languages, but was not as good in datasets that covered a broader spectrum of languages. To gain a more complete picture of the situation, we developed a "mostly-automated" method to collect Twitter messages that leverages the identity of the user to determine the language of messages, thus avoiding the direct application of LangID tools to construct a collection for evaluation of LangID. The dataset we constructed consists of 14178 messages across 65 different languages. Using this dataset, we showed that the accuracy of existing systems on Twitter messages is not as high as the accuracy reported in other domains. We also identified three systems that were relatively robust in the Twitter domain, including `langid.py`, a system based on our work in Chapter 5. Thereafter, we evaluated a number of techniques for adapting off-the-shelf LangID to a new domain. We found that bootstrapping additional in-domain training data was ineffective, and that the largest improvement, though still relatively small, was obtained by combining the predictions of three systems through a majority vote. Another small improvement was obtained by removing Twitter-specific content from messages. Finally, we provided a brief comparison to the "official" language predictions included in the Twitter API, and found that the API predictions are comparable in accuracy to state-of-the-art off-the-shelf language identifiers. One important observation is that a number of languages in use on Twitter appear to be unsupported by the language identifier of the Twitter API at the time of writing, underscoring the need for further research to broaden the scope and accuracy of LangID of Twitter messages.

## 8.1 Future Work

In this thesis, we addressed several aspects of generalized LangID that have so far been under-represented in the literature. We have made progress on dealing with variation in a language between different sources of text, and with LangID of documents that may contain text in more than one language. However, when it comes to eliminating assumptions made in LangID this is still just the tip of the proverbial iceberg; there are a number of areas that remain unexplored. In this section, we identify a number of such areas, as speculate on recent advances in machine learning and natural language processing that may yield insight into how to best tackle the issues.

### 8.1.1 Document Representation

One area that we focused extensive effort on in the course of this thesis is the design of a document representation that is suitable for LangID. As we discussed in Chapter 5, the key attribute that such a representation must have is that documents in the same language should, under the given representation, look similar regardless of the source they are drawn from, and at the same time be as different as possible from documents in any other language from any other source under the same representation. Our solution to this problem is a method of feature selection that takes into account both the language of a document and the source of text it is drawn from. This yields for each language a set of *language-indicative byte sequences*, and the distribution of these sequences in a document is the representation that we use. In other words, what we have identified are sequences that occur with roughly constant

relative frequency in any given language, regardless of the source of the document. This approach was shown to be effective in Chapter 5, and its simplicity has a number of advantages at the implementation level as it makes the derived classifier very simple and therefore very fast. That such an approach should even work is a novel result on its own, as it tells us something about language that is illustrated in Figure 5.2, in particular that 4-byte sequences are either strongly associated with source of text or with language; there is very limited middle ground in terms of features that are strongly associated with language and also with domain. However, even in the features strongly associated with language there is some variation in the strength of association with domain. Our approach so far has been to seek the features that are most strongly associated with language while being weakly associated with domain. The problem with this is that it discards potentially useful information from the slightly lower-ranked features. A simple method to demonstrate this would be to discard the top-$N$ features and use the next-$N$ features instead, which is likely to yield a classifier that is still above baseline (a similar argument was used by Joachims (1998) to argue for Support Vector Machines in Text Categorization). By only selecting a small number of features per language, our accuracy suffers in domains that have relatively short documents, such as TWITTER. A more conservative approach could attempt to model each feature in terms of both its per-language as well as its per-domain predictivity, in order to leverage information from a broader range of features.

Another issue with the feature selection that we presented is that features are scored independently, which leads to redundancy in the feature set. Very often, for

any byte 4-gram included, we also see all the 2-grams and 3-grams that the 4-gram can be broken down into, as well as 4-gram sequences with a 3-byte overlap. This independent scoring can cause problems, as we saw in the case of $\mathbb{IG}_{diff}$ versus $\mathbb{LD}$ (Section 5.5.3), where the global feature selection would select features that were only representative of a small number of languages. A more sophisticated feature selection method may be able to eliminate some of the redundancy in the feature set.

Another area that merits further analysis is the set of features that are discarded through this selection. Our hypothesis has been that the features we reject are more strongly associated with differences between domains than language differences. We may wish to revisit these features and examine them more closely in order to understand what sorts of differences these are, and evaluate whether it is possible to extract some additional information for LangID from the features we have currently rejected.

One aspect that is still relatively poorly understood is the effect of character encoding on LangID. In this thesis, we used data from a variety of encodings, but the majority of it was encoded in UTF8. As noted in Section 2.5.7, work to date has dealt with issues of encoding in various ways, but no real work has been done to investigate and compare different ways to handle encoding. As always, data availability is a problem, though this could partially be alleviated by transcoding existing document collections. There are still some outstanding issues with this: (1) it is not clear what encodings should be used in what languages, and furthermore in what proportions, and (2) the use of encoding may not be entirely independent of the document content – modern user-generated content is likely to be in a Unicode encoding, archived news

text may tend be in legacy encodings, and web content is likely to be in a mixture of both. Another point of comparison in handling multiple encodings for the same language is whether to treat all documents in the same language as a single class, or to treat each combination of encoding and language as a distinct class. Work to date has used both approaches (see Section 2.5.7), but no direct comparison has been made. The interaction between encoding and other aspects of document content is an interesting area well suited to further investigation.

### 8.1.2 Learning Algorithms

In Chapter 5, we argued that "simpler" learning algorithms were better suited to LangID because the class of decision boundaries that they are able to express more closely matches the "natural" decision boundaries of LangID problems due to their inherent property of invariance to document length. In terms of generalization error, simpler decision boundaries generally reduce variance at the expense of increased bias (underfitting), whereas more complex boundaries reduce bias at the expense of increasing variance (overfitting). However, in the specific case of LangID, the use of "simpler" learning algorithms, which produce relatively simple decision boundaries, has served to eliminate a specific bias induced by the average length of documents possibly being different in different languages. This is even more important in generalized LangID, where differences in length can easily result from differences between the sources of text used to train a classifier rather than actual differences between languages. As we have discussed in Section 8.1.1, our current approach to document representation is effective but discards potentially useful information. Similarly, the

use of "simpler" learning algorithms to enforce insensitivity to document length is crude but effective. However, in terms of the actual task, the key point is the elimination of the underlying bias rather than the specific implementation. The use of "simpler" learning algorithms has a disadvantage, in that it limits the classifier's ability to fit a more sophisticated decision boundary. As we eliminate biases through means such as the aforementioned cross-domain feature selection and enforcing insensitivity to document length, we expect variance to play a larger part in the overall generalization error. It therefore stands to reason that more sophisticated learning algorithms may be able to better control this variance, provided that the biases we have identified are similarly accounted for. Another notable similarity between the three systems examined in Chapter 4 is that all three use a generative model of language. As noted in Section 2.4, discriminative models have been considered in LangID and have been noted to be particularly effective in discriminating between closely-related languages, where their main advantage over generative models is in their inherent use of "negative evidence", identifying features that are strongly predictive of a document *not* being written in a specific language. However, there is a broader question of whether either generative or discriminative models are more suited to LangID in general, especially in the context of generalized LangID. Where LangID is treated as a supervised learning task, it may be the case that discriminative models are more effective, as is generally thought to be the case in any supervised learning task. However, most discriminative models cannot be easily extended to model more complex dependencies such as in the type of cross-domain or domain-agnostic learning required for generalized LangID. The naive solution, simply ignoring domain, risks overfitting

as the learning algorithm may learn discriminant features that distinguish between different sources of text rather than different languages. Nonetheless, given that our feature selection aims to eliminate this source of bias, it may still be worthwhile to further investigate discriminative training in the context of generalized LangID.

In our analysis in Chapter 5, we also found that VectorSpaceModel generally underperformed RanklistModel and LikelihoodModel. Error analysis in Section 5.6.6 highlighted some interesting observations that may be interesting to explore further. In Section 5.6.6, we speculated that the poor accuracy of VectorSpaceModel may be due to the classifier still learning source-specific clusters for each language despite the use of $\mathbb{LD}$ features, with closely-related languages appearing as interceding clusters between other clusters for the same language. We did not test this hypothesis in this thesis, but it would be of interest to do so in future work. If this is the case, it would suggest that under VectorSpaceModel is generally less suited to generalized LangID than LikelihoodModel and RanklistModel as it is more sensitive to intra-lingual differences between sources, or perhaps that an alternative feature selection to $\mathbb{LD}$ is needed to work with VectorSpaceModel.

### 8.1.3 Closely-related Languages

In Section 2.5.6, we identified many sets of closely-related languages that had been investigated in the literature from the perspective of classifying documents according to their language/dialect/variety. Work to date has generally been able to separate documents within a set of closely-related languages to a high degree of accuracy, but little work has been done in integrating the results back into a more generalized

LangID system. Results from the VarDial discriminating between similar languages shared task (Zampieri *et al.* to appear) suggest that a viable approach is a two-level hierarchical classification, where the group of languages is identified first, followed by a per-group classifier. Open issues in this area surround the details such as how to automatically determine the sets of languages for which a second-level classifier is required. For this, there is potential to draw on extensive work in linguistics on language phylogeny.

### 8.1.4 Number of Languages

The datasets we compiled for this thesis cover 145 languages, which is a respectable number compared to previous work (Table 2.2) but is also a far cry from the 7000+ languages cataloged as living by the Ethnologue (Gordon 2005). Some work has attempted to expand the set of languages covered by a single system into the thousands (Brown 2013), but our evaluation in Chapter 7 of `whatlang`, the off-the-shelf implementation of this method, found that the system suffered in terms of accuracy because it predicted many languages that were not present in the test data. Collecting sufficient training data for many of the "long-tail" languages is an ongoing problem, though there are a number of efforts to compile data from a variety of sources, such as ODIN (Xia *et al.* 2010a), as well as the Crúbadán project (Scannell 2007). However, these are not efforts to build a broad-scope LangID system, but rather efforts to collect data for under-resourced languages and are one of the main *consumers* of such a broad-scope LangID system. In developing broad-scope LangID, an important question to be answered is how much data do we need to model a language? The

results in this thesis suggest that there may not be a simple answer to this question; accuracy varies according to the number and variety of other languages modeled, as well as in the diversity of the data available to model a specific language. In modeling lower-density languages on the basis of limited amounts of text in Section 6.6.1, we still saw that having multiple sources for the same language was useful, and that the utility was not just due to the increased amount of training data.

### 8.1.5 "Unseen" Languages

Many of the LangID systems we have examined will always make a prediction from the closed set of languages they have training data for, and as we saw in Chapter 7 this can be problematic when a target domain includes languages for which the LangID system does not have training data, because the precision of prediction in other languages will be adversely affected. This suggests that developing a language identifier that is able to identify when a document is unlikely to belong to any of the languages for which we have training data. We discussed a number of approaches for this in Section 2.5.3, where we found that this problem is typically tackled by thresholding on a confidence score produced by a classifier, a technique that is used to reasonable effect in `ChromeCLD`, an off-the-shelf language identifier. However, another possible approach that has yet to be explored in the literature is the extension of the generative mixture models to "unknown" LangID. In Chapter 6, we discussed the analogy between topic modeling and LangID over multilingual documents, and showed how generative mixture models have been used to tackle both problems. An issue that has plagued topic modeling is the need to specify the number of topics in

advance. A recent development in topic modeling is the use of non-parametric mixture models such based on a Hierarchical Dirichlet Process (Teh *et al.* 2006a), which provide the ability to identify "new" topics. Similar reasoning could likely be applied in order to identify "new" languages.

### 8.1.6 Multilingual Documents

The method that we proposed for LangID of multilingual documents in Chapter 6 has at its core a model that is very similar to that used in LDA for topic modeling. One aspect of this model is that it maintains an explicit mapping between tokens and labels, which is inferred through the use of Gibbs sampling. This approach has been popular because it is relatively simple to implement and produces reasonable results, but has the disadvantage of being computationally intensive (and inherently difficult to parallelize) due to the iterative sampling required. It may be possible to reduce the computational cost through alternative methods of inference such as collapsed variational inference (Teh *et al.* 2006b). Alternatively, since the individual topic assignments are not directly useful in terms of identifying languages present and their relative proportions in a multilingual document, it may be possible to integrate the mapping between tokens and languages out entirely, which should result in a Dirichlet posterior that we can sample from.

### 8.1.7 Text Segmentation by Language

In Chapter 6, we evaluated our method for LangID in multilingual documents against `SegLang` (Yamaguchi and Tanaka-Ishii 2012), a system that segments mul-

tilingual documents into monolingual segments, and found that `SegLang` achieved relatively poor precision due to a tendency to over-segment a document. One of the motivations given for developing multilingual LangID is to answer a call by King and Abney (2013) for exactly such a method in order to complement their research on word-level LangID, which assumes that the languages present in a document are known in advance. The combination of our method and the method of King and Abney (2013) could thus be used to segment an arbitrary document by language, and could be compared to that of Yamaguchi and Tanaka-Ishii (2012) in the context of constructing corpora for low-density languages using the web.

Another approach to this task may be to leverage work on changepoint detection, which aims to identify abrupt changes in the generative parameters of sequential data. Such a model appears to be a good fit for the problem of detecting and segmenting text written in one language concatenated to text written in another language, and so it may be possible to apply Bayesian techniques for online changepoint detection (Adams and MacKay 2007) to the task of text segmentation by language. Text segmentation by language also bears some similarity to the problem of finding subtopic paragraphs using distributional features (Hearst 1997), and could be tackled with similar algorithms.

### 8.1.8 LangID of Short Texts

One of the challenges we identified in LangID of Twitter messages (Chapter 7) is that the short length of the messages provides relatively little data on which to base a classification. This is perhaps one of the vulnerabilities of the method we

describe in Chapter 5, which through feature selection discards a large part of the message. We found that TWITTER was one of the few domains in which increasing the number of features selected per-language from 50 to 300 produced an appreciable increase in LangID accuracy. As discussed in Section 2.5.5, LangID of short texts has broad applicability beyond social medial messages, to tasks such as LangID of search engine queries or perhaps also word-level LangID in multilingual documents. The challenge is thus to make full(er) use of the information available in a short text. Nakatani (2012) proposes the use of "all substring" features (Okanohara and Tsujii 2009) for this purpose, which are implemented in the `LDIG` system which we showed to be highly effective in the limited set of languages it supports in Chapter 7. Future work in this area could also draw on related work in language models, particularly on smoothing techniques such as Kneser-Ney interpolation (Chen and Goodman 1999), to better estimate the probability of $n$-gram sequences not seen in the training data.

### 8.1.9 Contextual information for LangID

Our focus in this thesis has been on LangID of documents based solely on their textual content, without using any information about the source of the text. In Chapter 4, we showed that there are characteristics of languages that are consistent between different sources of text, and that can be exploited to train a generalized language identifier. However, as we discussed in Section 2.5.8, there are often domain-specific contextual features that can be useful for identifying the language of a text. These features can be in the form of explicit metadata available only in a specific domain, such as geolocation information or user identity, or it can be in the form of domain-

specific characteristics, which we have explicitly tried to reject in constructing our robust document representation. Since we have seen that LangID systems trained on in-domain data generally outperform systems trained on generalized data, a different approach to generalized LangID is to have a collection of domain-specific LangID systems, and generalize their predictions, either through an ensemble approach similar to the one we used in Chapter 7, or perhaps through hierarchical classification, where a document is first classified by domain, and thereafter a domain-specific classifier is applied. Another approach is to model this as a multi-view learning problem, where we model the text content of a document as one view, and the document-external context as an alternative view.

In this thesis, we have only had the opportunity to explore domain-specific LangID in Twitter (Chapter 7). However, there are a number of areas where it may be interesting to further explore domain-specific approaches. One such area is in dealing with machine-translated text: Machine-translated documents are known to have different characteristics from "natural" documents (Baroni and Bernardini 2006), so it is plausible that we can improve the accuracy of LangID by taking advantage of these characteristics when a document is known to be machine-translated. There is also demand for robust NLP tools for historical documents resulting from digital humanities research and cultural heritage projects, as languages are known to change and develop over time, and these changes can be detected by automatic methods and used to date text (Niculae *et al.* 2014). The work in this thesis could perhaps be adapted to date-sensitive LangID, by attempting to identify characteristics of a language that have remained relatively invariant over time.

The assumption that no unlabeled documents from the same source are available reflects a common use case for LangID, but limits the applicability of techniques from general areas such as domain adaptation and robust learning. In many cases, additional unlabeled text in the target domain may be readily available, such as in digital forensics, e-discovery of legal documents, or linguistic corpus creation from web data. In such settings, techniques such as structural correspondence learning, co-training, importance weighting and model regularization may further improve LangID accuracy by taking domain-specific information into account.

### 8.1.10 Standardized LangID evaluation

In Section 2.5.9, we discussed how objective comparison of different methods for LangID is difficult due to the different data that authors have used for training and evaluation, and looked at the characteristics of some of the more popular sources. As we hope to have convinced the reader in the course of this thesis, LangID is an interesting and nuanced problem and could serve as an excellent test-case for future work on the application of learning techniques to text, including subproblems such as representation learning, semi-supervised learning, multi-view learning and domain adaptation. For this to be possible, there is an underlying need for standardized corpora for evaluation, and standardized metrics to evaluate against. From our discussion in Section 2.2.4, it is clear that there is some diversity in metrics used and the aspects of the problem they capture. Future work in this area should work towards establishing fully-reproducible results, through the release of training and test data and accompanying software to reproduce experimental results where possible. One

possible pathway to facilitate this is by separating the work of data generation from the problem solving aspects through the organization of shared tasks (see Table 2.4 on page 80). Shared tasks generate published training and test data and use standardized evaluation metrics to allow comparability between participants, and have been very successful in driving participation in other research areas (e.g. TREC for information retrieval or SemEval for computational semantic analysis). Similar shared tasks have started exploring aspects of the LangID problem, including discriminating between similar languages (Zampieri *et al.* 2014) and multilingual documents (Baldwin and Lui 2010b). A general, broad-coverage LangID shared task has yet to be proposed, and would be an excellent means to encourage further research in LangID. One challenge in setting up such a shared task is in constructing an evaluation that would allow us to compare different systems on a broad range of input distributions.

## 8.2 Closing Remarks

In this thesis, we explored *generalized* language identification, the problem of determining what natural language a document (or part there of) is written in. In the course of our discussion, we have shown that this is a problem that has attracted interest from a diverse variety of research communities. Furthermore, it is a nuanced problem with many different aspects and open issues that hold great promise for future research. We have only been able to tackle a small portion of the open questions in this thesis, but nonetheless we hope that the results and discussion have provided the reader with some insight into the task, and that this thesis may serve as a reference to motivate, encourage, and guide future work in LangID.

# Bibliography


Abney, Steven, and Steven Bird. 2010. The human language project: building a universal corpus of the world's languages. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, 88–97, Los Angeles, USA.

Adams, Gary, and Philip Resnik. 1997. A language identification application built on the Java client/server platform. In *Proceedings of the ACL/EACL'97 Workshop on From Research to Commercial Applications: Making NLP Work in Practice*, 43–47, Madrid, Spain.

Adams, Ryan Prescott, and David JC MacKay. 2007. Bayesian online changepoint detection. Technical report, University of Cambridge, Cambridge, UK.

Alex, Beatrice. 2006. Integrating language knowledge resources to extend the English inclusion classifier to a new language. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, 2431–2436, Genoa, Italy.

——, Amit Dubey, and Frank Keller. 2007. Using foreign inclusion detection to improve parsing performance. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2007 (EMNLP-CoNLL 2007)*, 151–160, Prague, Czech Republic.

Amine, Abdelmalek, Zakaria Elberrichi, and Michel Simonet. 2010. Automatic language identification: An alternative unsupervised approach using a new hybrid algorithm. *International Journal of Computer Science and Applications* 7.94–107.

Armstrong-Warwick, Susan, S Thompson, David McKelvie, and Dominique Petitpierre. 1994. Data in your language: the ECI multilingual corpus 1. In *Proceedings of International Workshop on Sharable Natural Language Resources*, 97–106, Ikoma, Japan.

Arnold, Andrew, Ramesh Nallapati, and William W Cohen. 2007. A comparative study of methods for transductive transfer learning. In *Proceedings of the Seventh IEEE International Conference on Data Mining (Workshops)*, 77–82, Omaha, USA.

Artemenko, Olga, Thomas Mandl, Margaryta Shramko, and Christa Womser-Hacker. 2006. Evaluation of a language identification system for mono- and multilingual text documents. In *Proceedings of the 2006 ACM symposium on Applied computing (SAC 06)*, 859–860, New York, USA.

Baldwin, Timothy, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang. 2013. How noisy social media text, how diffrnt social media sources? In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP 2013)*, 356–364, Nagoya, Japan.

——, and Marco Lui. 2010a. Language identification: The long and the short of the matter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, 229–237, Los Angeles, USA.

——, and ——. 2010b. Multilingual language identification: ALTW 2010 shared task dataset. In *Proceedings of the Australasian Language Technology Workshop 2010 (ALTW 2010)*, 5–7, Melbourne, Australia.

Baroni, Marco, and Silvia Bernardini. 2006. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing* 21.259–274.

Baykan, Eda, Monika Henzinger, and Ingmar Weber. 2008. Web page language identification based on urls. In *Proceedings of the 34th International Conference on Very Large Data Bases (VLDB 2008)*, 176–187, Auckland, New Zealand.

Beesley, Kenneth R. 1988. Language Identifier: A computer program for automatic natural-language identification of on-line text. In *Languages at Crossroads: Proceedings of the 29th Annual Conference of the American Translators Association*, 47–54, Seattle, USA.

Bergsma, Shane, Paul McNamee, Mossaab Bagdouri, Clayton Fink, and Theresa Wilson. 2012. Language identification for creating language-specific Twitter collections. In *Proceedings the Second Workshop on Language in Social Media (LSM2012)*, 65–74, Montréal, Canada.

Biemann, Chris, and Sven Teresniak. 2005. Disentangling from babylonian confusion – unsupervised language identification. In *Proceedings of 6th International Conference in Computational Linguistics and Intelligent Text Processing (CICLing 2005)*, 773–784, Mexico City.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research* 3.993–1022.

Bontcheva, Kalina, Leon Derczynski, Adam Funk, Mark A. Greenwood, Diana Maynard, and Niraj Aswani. 2013. TwitIE: An open-source information extraction pipeline for microblog text. In *Proceedings of Recent Advances in Natural Language Processing (RANLP 2013)*, 83–90, Hissar, Buglaria.

Bosca, Alessio, and Luca Dini. 2010. Language identification strategies for cross language information retrieval. In *Working notes for LogCLEF2010: the CLEF 2010 Multilingual Logfile Analysis Track*, Padua, Italy.

Botha, Gerrit Reinier, and Etienne Barnard. 2012. Factors that affect the accuracy of text-based language identification. *Computer Speech & Language* 26.307–320.

Brown, Ralf. 2013. Selecting and weighting n-grams to identify 1100 languages. In *Proceedings of the 16th International Conference on Text, Speech and Dialogue (TSD 2013)*, 475–483, Plzeň, Czech Republic.

——. 2014. Non-linear mapping for improved identification of 1300+ languages. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 627–632, Doha, Qatar.

Brown, Ralf D. 2012. Finding and identifying text in 900+ languages. *Digital Investigation* 9.S34–S43.

Carter, Simon, Manos Tsagkias, and Wouter Weerkamp. 2011. Semi-supervised priors for microblog language identification. In *Proceedings of the 11th Dutch-Belgian Information Retrieval Workshop (DIR 2011)*, 12–15, Amsterdam, Netherlands.

——, Wouter Weerkamp, and Manos Tsagkias. 2013. Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text. *Language Resources and Evaluation* 47.1–21.

Cavnar, William B., and John M. Trenkle. 1994. *N*-gram based text categorization. In *Proceedings of the Third Symposium on Document Analysis and Information Retrieval*, 161–175, Las Vegas, USA.

Ceylan, Hakan, and Yookyung Kim. 2009. Language identification of search engine queries. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 1066–1074, Singapore.

Chen, Stanley F, and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language* 13.359–393.

Chew, Yew Choong, Yoshiki Mikami, and Robin Lee Nagano. 2011. Language identification of web pages based on improved $n$-gram algorithm. *International Journal of Computer Science Issues* 8.47–58.

Choong, Chew Y., Yoshiki Mikami, C. A. Marasinghe, and S. T. Nandasara. 2009. Optimizing $n$-gram order of an $n$-gram based language identification algorithm for 68 written languages. *International Journal on Advances in ICT for Emerging Regions (ICTer)* 02.21–28.

ClueWeb09, 2009. The ClueWeb09 dataset. `http://lemurproject.org/clueweb09/`.

Cole, Ronald, Joseph Mariani, Hans Uszkoreit, Giovanni Battista Varile, Annie Zaenen, Antonio Zampolli, and Victor Zue (eds.) 1997. Cambridge, UK: Cambridge University Press.

Combrinck, Hendrik Petrus, and E.C. Botha. 1995. Text-based automatic language identification. In *Proceedings of the 6th Annual Symposium of the Pattern Recognition Association of South Africa*, Gauteng, South Africa.

Constable, Peter, and Gary Simons. 2000. Language identification and IT: Addressing problems of linguistic diversity on a global scale. SIL Electronic Working Papers 2000-001, SIL International, Dallas, USA.

Cook, Paul, and Marco Lui. 2012. langid.py for better language modelling. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, 107–112, Dunedin, New Zealand.

Cover, Thomas M., and Joy A. Thomas. 2006. *Elements of Information Theory (2nd ed.)*. New York, USA: Wiley.

Cowie, Jim, Yevgeny Ludovik, and Ron Zacharski. 1999. Language recognition for mono- and multi-lingual documents. In *Proceedings of the Vextal Conference*, 209–214, Venice, Italy.

Culotta, Aron. 2010. Towards detecting influenza epidemics by analyzing Twitter messages. In *Proceedings of the first KDD Workshop on Social Media Analytics*, 115–122, Washington, USA.

Da Silva, Joaquim, and Gabriel Lopes. 2006. Identification of document language is not yet a completely solved problem. In *Proceedings of the International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, 212–212, Sydney, Australia.

Daumé III, Hal. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, 256–263, Prague, Czech Republic.

——, and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research* 26.101–126.

Debole, Franca, and Fabrizio Sebastiani. 2005. An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and technology* 56.584–596.

Deneubourg, Jean-Louis, Simon Goss, Nigel Franks, Ana Sendova-Franks, Claire Detrain, and Laeticia Chrétien. 1990. The dynamics of collective sorting: robot-like ants and ant-like robots. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior*, 356–363, Paris, France.

Derczynski, Leon, Diana Maynard, Niraj Aswani, and Kalina Bontcheva. 2013. Microblog-genre noise and impact on semantic annotation accuracy. In *24th ACM Conference on Hypertext and Social Media*, 21–30, Paris, France.

Dietterich, Thomas G. 2002. Ensemble learning. In *The Handbook of Brain Theory and Neural Networks*, ed. by Michael A. Arbib, 405–408. Cambridge, USA: MIT Press.

Diwersy, Sascha, Stefan Evert, and Stella Neumann. 2014. A weakly supervised multivariate approach to the study of language variation. In *Aggregating Dialectology, Typology, and Register Analysis. Linguistic Variation in Text and Speech*, ed. by Benedikt Szmrecsanyi and Bernhard Wälchli. Berlin: De Gruyter.

Dueire Lins, Rafael, and Paulo Gonçalves. 2004. Automatic language identification of written texts. In *Proceedings of the 2004 ACM Symposium on Applied Computing (SAC 2004)*, 1128–1133, Nicosia, Cyprus.

Dunning, Ted. 1994. Statistical identification of language. Technical Report MCCS 940-273, Computing Research Laboratory, New Mexico State University.

Eisenstein, Jacob. 2013. What to do about bad language on the internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, 359–369, Atlanta, USA.

Elfardy, Heba, and Mona Diab. 2013. Sentence level dialect identification in Arabic. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 456–461, Sofia, Bulgaria.

Elworthy, David. 1998. Language identification with confidence limits. In *Proceedings of the 6th Annual Workshop on Very Large Corpora*, 94–101, Montreal, Canada.

Evgeniou, Theodoros, and Massimiliano Pontil. 2004. Regularized multi–task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 109–117, Seattle, USA.

Forman, George. 2003. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research* 3.1289–1305.

Gao, Sheng, Wen Wu, Chin-Hui Lee, and Tat-Seng Chua. 2004. A MFOM learning approach to robust multiclass multi-label text categorization. In *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada.

Ghamrawi, Nadia, and Andrew McCallum. 2005. Collective multi-label classification. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management (CIKM 2005)*, 195–200, Bremen, Germany.

Ghani, Rayid, Rosie Jones, and Dunja Mladenic. 2004. Building minority language corpora by learning to generate web search queries. *Knowledge and Information Systems* 7.56–83.

Giguet, Emmanuel. 1995. Categorisation according to language: A step toward combining linguistic knowledge and statistical learning. In *Proceedings of the 4th International Workshop on Parsing Technologies (IWPT-1995)*, Prague, Czech Republic.

Giles, Jim. 2005. Internet encyclopaedias go head to head. *Nature* 438.900–901.

Gimpel, Kevin, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 42–47, Portland, USA.

Giwa, Oluwapelumi, and Marelie H. Davel. 2013. N-gram based language identification of individual words. In *Proceedings of the 24th Annual Symposium of the Pattern Recognition Association of South Africa*, 15–22, Johannesburg, South Africa.

Gold, E. Mark. 1967. Language identification in the limit. *Information and Control* 5.447–474.

Goldszmidt, Moises, Marc Najork, and Stelios Paparizos. 2013. Bootstrapping language identifiers for short colloquial postings. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECMLPKDD 2013)*, 95–111, Prague, Czech Republic.

Goodman, Kenneth S (ed.) 1973. *Miscue analysis: applications to reading instruction.* Urbana, USA: ERIC Clearinghouse on Reading and Communication Skills.

Gordon, Raymond G. 2005. *Ethnologue: Languages of the World.* Dallas, USA: SIL International.

Gottron, Thomas, and Nedim Lipka. 2010. A comparison of language identification approaches on short, query-style texts. In *Proceedings of Advances in Information Retrieval, 32nd European Conference on IR Research (ECIR 2010)*, 611–614, Milton Keynes, UK.

Grefenstette, Gregory. 1995. Comparing two language identification schemes. In *Proceedings of Analisi Statistica dei Dati Testuali (JADT)*, 263–268, Rome, Italy.

—— (ed.) 1998. *The problem of cross-language information retrieval.* Berlin, Germany: Springer.

Griffiths, Thomas. 2002. Gibbs sampling in the generative model of latent Dirichlet allocation. Technical report, Stanford University.

Griffiths, Thomas L, and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences of the United States of America* 101.5228–5235.

Grothe, Lena, Ernesto William De Luca, and Andreas Nürnberger. 2008. A comparative study on language identification methods. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, 980–985, Marrakesh, Morocco.

Gupta, Suhit, Gail Kaiser, David Neistadt, and Peter Grimm. 2003. DOM-based content extraction of HTML documents. In *Proceedings of the 12th International Conference on the World Wide Web (WWW 2003)*, 207–214, Budapest, Hungary.

Hakkinen, Juha, and Jilel Tian. 2001. *n*-gram and decision tree based language identification for written words. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding, 2001. ASRU '01.*, 335–338, Madonna di Campiglio, Italy.

Hammarström, Harald. 2007. A fine-grained model for language identification. In *Proceedings of Improving Non English Web Searching (iNEWS' 07)*, 14–20, Amsterdam, Netherlands.

Han, Bo, Paul Cook, and Timothy Baldwin. 2013. Lexical normalisation of short text messages. *ACM Transactions on Intelligent Systems and Technology* 4.5:1–5:27.

——, ——, and ——. 2014a. Text-based Twitter user geolocation prediction. *Journal of Artificial Intelligence Research* 49.451–500.

——, ——, and ——. 2014b. Text-based Twitter user geolocation prediction. *Journal of Artificial Intelligence Research (JAIR)* 49.451–500.

——, Marco Lui, and Timothy Baldwin. 2011. Melbourne language group microblog track report. In *Proceedings of the 20th Text REtrieval Conference (TREC 2011)*. NIST Special Publication: SP 500-295.

Harris, Zellig S. 1970. Distributional structure. In *Papers in Structural and Transformational Linguistics*, Formal Linguistics Series, 775–794. Dordrecht, Netherlands: Springer Netherlands.

Hearst, Marti A. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics* 23.33–64.

Hersh, William, Chris Buckley, TJ Leone, and David Hickam. 1994. OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *Proceedings of 17th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, 192–201, Dublin, Ireland.

House, Arthur S, and Edward P Neuburg. 1977. Toward automatic identification of the language of an utterance. *The Journal of the Acoustical Society of America* 62.708.

Huang, Chu-Ren, and Lung-Hao Lee. 2008. Contrastive approach towards text source classification based on top-bag-of-word similarity. In *Proceedings of the 22nd Pacific Asia Conference on Language, Information and Computation*, 404–410, Cebu City, Philippines.

Hughes, Baden, Timothy Baldwin, Steven Bird, Jeremy Nicholson, and Andrew MacKinlay. 2006. Reconsidering language identification for written language resources. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, 485–488, Genoa, Italy.

Ingle, Norman. 1976. A language identification table. *The Incorporated Linguist* 15.98–101.

Jiang, Jing. 2009. Multi-task transfer learning for weakly-supervised relation extraction. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, 1012–1020, Singapore.

——, and ChengXiang Zhai. 2006. Exploiting domain structure for named entity recognition. In *Proceedings of COLING/ACL 2006*, 74–81, Sydney, Australia.

Joachims, Thorsten. 1998. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, 137–142, Chemnitz, Germany.

Johnson, Stephen. 1993. Solving the problem of language recognition. Technical report, School of Computer Studies, University of Leeds.

Jortin, J., and J.L. Clerc. 1808. *The life of Erasmus*, volume 1. London, UK: Richard Taylor and Co.

Kay, Martin. 1997. Multilinguality. In (Cole *et al.* 1997), 281–284.

Kikui, Genitiro. 1996. Identifying the coding system and language of on-line documents on the internet. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*, 652–657, Kyoto, Japan.

Kilgarriff, Adam. 2001. Comparing corpora. *International Journal of Corpus Linguistics* 6.97–133.

King, Ben, and Steven Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1110–1119, Atlanta, Georgia.

Knuth, Donald E. 1998. *The Art of Computer Programming*, volume 2. Reading, USA: Addison-Wesley Publishing Company.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the Tenth Machine Translation Summit (MT Summit X)*, Phuket, Thailand.

Konstantopoulos, Stasinos. 2007. What's in a name? In *Proceedings of the 2007 Conference on Recent Advances in Natural Language Processing (RANLP-07)*, Borovets, Bulgaria.

Kralisch, Anett, and Thomas Mandl. 2006. Barriers to information access across languages on the internet: Network and language effects. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, volume 3, p. 54b, Kauai, USA.

Krishnamurthy, Balachander, Phillipa Gill, and Martin Arlitt. 2008. A few chirps about Twitter. In *Proceedings of the First Workshop on Online Social Networks (WOSN 2008)*, 19–24, Seattle, USA.

Kruengkrai, Canasai, Prapass Srichaivattana, Virach Sornlertlamvanich, and Hitoshi Isahara. 2005. Language identification based on string kernels. In *Proceedings of the 5th International Symposium on Communications and Information Technologies (ISCIT-2005)*, 896–899, Beijing, China.

Lampos, Vasileios, Tijl De Bie, and Nello Cristianini. 2010. Flu Detector – tracking epidemics on Twitter. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2010)*, 599–602, Barcelona, Spain.

Lee, Lillian. 1999. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 25–32, College Park, USA.

Lewis, David D., 1997. The Reuters-21578 data set. Available at `http://www.daviddlewis.com/resources/testcollections/reuters21578/`.

Lewis, M. Paul, Gary F. Simons, and Charles D. Fennig (eds.). 2014. *Ethnologue: Languages of the World, Seventeenth edition.* Dallas, USA: SIL International. Online version: `http://www.ethnologue.com`.

Lewis, William D, and Fei Xia. 2010. Developing ODIN: A multilingual repository of annotated language data for hundreds of the world's languages. *Literary and Linguistic Computing* 25.303–319.

Ling, Wang, Chris Dyer, Alan W Black, and Isabel Trancoso. 2013. Paraphrasing 4 microblog normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 73–84, Seattle, USA.

Ljubešić, Nikola, and Filip Klubička. 2014. {bs,hr,sr}WaC – web corpora of Bosnian, Croatian and Serbian. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, 29–35, Gothenburg, Sweden. Association for Computational Linguistics.

Ljubešić, Nikola, Nives Mikelić, and Damir Boras. 2007. Language identification: how to distinguish similar languages? In *Proceedings of the 29th International Conference on Information Technology Interfaces*, 541–546, Cavtat, Croatia.

Lui, Marco, and Timothy Baldwin. 2011. Cross-domain feature selection for language identification. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, 553–561, Chiang Mai, Thailand.

——, and ——. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012) Demo Session*, 25–30, Jeju, Republic of Korea.

——, and ——. 2014. Accurate language identification of Twitter messages. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, 17–25, Gothenburg, Sweden. Association for Computational Linguistics.

——, and Paul Cook. 2013. Classifying english documents by national dialect. In *Proceedings of the Australasian Language Technology Association Workshop 2013 (ALTA 2013)*, 5–15, Brisbane, Australia.

——, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics* 2.27–40.

Majliš, Martin. 2012. Yet another language identifier. In *Proceedings of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 46–54, Avignon, France.

Mandl, Thomas, Margaryta Shramko, Olga Tartakovski, and Christa Womser-Hacker. 2006. Language identification in multi-lingual web-documents. In *Proceedings of the 11th International Conference on Applications of Natural Language to Information Systems (NLDB 2006)*, 153–163, Klagenfurt, Austria.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics* 19.313–330.

Martins, Bruno, and Mário J. Silva. 2005. Language identification in web pages. In *Proceedings of the 2005 ACM symposium on Applied computing*, 764–768, Santa Fe, USA.

Mayer, Uwe F. 2012. Bootstrapped language identification for multi-site internet domains. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2012)*, 579–585, Beijing, China.

McCallum, Andrew, and Kamal Nigam. 1998. A comparison of event models for Naive Bayes text classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*, Available as Technical Report WS–98–05, AAAI Press., Madison, USA.

McCallum, Andrew Kachites. 1999. Multi-label text classification with a mixutre model trained by EM. In *Proceedings of AAAI'99 Workshop on Text Learning*, 1–7, Orlando, USA.

McNamee, Paul. 2005. Language identification: a solved problem suitable for undergraduate instruction. *Journal of Computing Sciences in Colleges* 20.94–101.

——, and James Mayfield. 2004. Character n-gram tokenization for european language text retrieval. *Information Retrieval* 7.73–97.

Milne, Rachel Mary, Richard A. O'Keefe, and Andrew Trotman. 2012. A study in language identification. In *Proceedings of the Seventeenth Australasian Document Computing Symposium*, 88–95, Dunedin, New Zealand.

Mitchell, Tom M. 1997. *Machine Learning.* New York, USA: McGraw-Hill.

Moulton, W.G. 1975. *A Linguistic Guide to Language Learning.* New York, USA: Modern Language Association of America.

Murthy, Kavi Narayana, and G. Bharadwaja Kumar. 2006. Language identification from small text samples. *Journal of Quantitative Linguistics* 13.57–80.

Muthusamy, Yeshwant K, and A Lawrence Spitz. 1997. Automatic language identification. In (Cole *et al.* 1997), 314–317.

Nakatani, Shuyo, 2010a. Language detection library for Java. `http://code.google.com/p/language-detection/`. Retrieved on 21/06/2013.

——, 2010b. Language detection library. Slides. `http://www.slideshare.net/shuyo/language-detection-library-for-java`. Retrieved on 21/06/2013.

——, 2012. Short text language detection with infinity-gram. Blog post. `http://shuyo.wordpress.com/2012/05/17/short-text-language-detection-with-infinity-gram/`. Retrieved on 21/06/2013.

Newman, Patricia. 1987. Foreign language identification: First step in the translation process. Technical report, Sandia National Labs., Albuquerque, NM (USA).

Ng, Choon-Ching, and Ali Selamat. 2011. Improving language identification of web page using optimum profile. *Communications in Computer and Information Science* 180.157–166.

Nguyen, Dong, and A. Seza Dogruoz. 2013. Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, 857–862, Seattle, USA.

Niculae, Vlad, Marcos Zampieri, Liviu Dinu, and Alina Maria Ciobanu. 2014. Temporal text ranking and automatic dating of texts. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, 17–21, Gothenburg, Sweden. Association for Computational Linguistics.

Nie, Jian-Yun. 2010. *Cross-language information retrieval.* San Rafael, USA: Morgan and Claypool Publishers.

——, Michel Simard, Pierre Isabelle, and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *Proceedings of 22nd International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, 74–81, Berkeley, USA.

Okanohara, Daisuke, and Jun'ichi Tsujii. 2009. Text categorization with all substring features. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, 838–846, Miami, USA.

Padró, Munsta, and Lluís Padró. 2004. Comparing methods for language identification. In *Proceedings the XX Congreso de la Sociedad Española para el Procesamiento del Lenguage Natural (SEPLN' 04)*, 155–162, Barcelona, Spain.

Pan, Sinno Jialin, and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22.1345–1359.

Peirsman, Yves, Dirk Geeraerts, and Dirk Speelman. 2010. The automatic identification of lexical variation between language varieties. *Natural Language Engineering* 16.469–491.

Peng, Fuchun, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, 562–568, Geneva, Switzerland.

Petrović, Sasa, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to Twitter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, 181–189, Los Angeles, USA.

Pienaar, Wikus, and Dirk Snyman. 2010. Spelling checker-based language identification for the eleven official South African languages. In *Proceedings of the 21st Annual Symposium of the Pattern Recognition Association of South Africa*, 219–224, Stellenbosch, South Africa.

Poutsma, Arjen. 2002. Applying Monte Carlo techniques to language identification. *Language and Computers* 45.179–189.

Prager, John M. 1999a. Linguini: language identification for multilingual documents. In *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences (HICSS-32)*, Maui, USA.

—— 1999b. Linguini: Language identification for multilingual documents. *Journal of Management Information Systems* 16.71–101.

Quinlan, J.R. 1986. Induction of decision trees. *Machine Learning* 1.81–106.

Ramage, Daniel, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, 248–256, Singapore.

Ranaivo-Malancon, Bali. 2006. Automatic identification of close languages – case study: Malay and Indonesian. *ECTI Transaction on Computer and Information Technology* 2.126–134.

Rehurek, Radim, and Milan Kolkus. 2009. Language identification on the web: Extending the dictionary method. In *Proceedings of the 10th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2009)*, 357–368, Mexico City, Mexico.

Resnik, Philip. 1999. Mining the Web for bilingual text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 527–534, College Park, USA.

Rocchio, Joseph John Jr. 1971. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, ed. by Gerard Salton, Prentice-Hall Series in Automatic Computation, chapter 14, 313–323. Englewood Cliffs, USA: Prentice-Hall.

Roomann-Kurrik, Arne, 2013. Introducing new metadata for tweets. Blog post. `https://dev.twitter.com/blog/introducing-new-metadata-for-tweets`. Retrieved on 29/07/2014.

Sagiroglu, Seref, Uraz Yavanoglu, and Esra Nergis Guven. 2007. Web based machine learning for language identification and translation. In *Proceedings of the Sixth International Conference on Machine Learning and Applications*, 280–285, Cincinati, USA.

Sakaki, Takeshi, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on the World Wide Web (WWW 2010)*, 851–860, Raleigh, USA.

Scannell, Kevin P. 2007. The Crúbadán Project: Corpus building for under-resourced languages. In *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, 5–15, Louvain-la-Neuve, Belgium.

Schapire, R.E., and Y. Singer. 2000. BoosTexter: A boosting-based system for text categorization. *Machine Learning* 39.135–168.

Scheelen, Frank, 2003. *libtextcat*. Software available at `http://software.wise-guys.nl/libtextcat/`.

Sebastiani, Fabrizio. 2002. Machine learning in automated text categorization. *ACM computing surveys (CSUR)* 34.1–47.

Sibun, Penelope, and Jeffrey C. Reynar. 1996. Language determination: Examining the issues. In *Proceedings of the 5th Annual Symposium on Document Analysis and Information Retrieval*, 125–135, Las Vegas, USA.

——, and A Lawrence Spitz. 1994. Language determination: Natural language processing from scanned document images. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, 15–21, Stuttgart, Germany.

Simões, Alberto, José João Almeida, and Simon D. Byers. 2014. Language identification: a neural network approach. In *Proceedings of the 3rd Symposium on Languages, Applications and Technologies (SLATE 2014)*, 251–265, Bragança, Portugal.

Singh, Anil Kumar. 2006. Study of some distance measures for language and encoding identification. In *Proceedings of the Workshop on Linguistic Distances*, 63–72, Sydney, Australia.

Sites, Dick, 2013a. Cld2fullversion. online manuscript. available at `http://code.google.com/p/cld2/wiki/CLD2FullVersion`.

——, 2013b. *Compact Language Detector 2*. Software available at `http://code.google.com/p/cld2/`.

Solorio, Thamar, Elizabeth Blair, Suraj Maharjan, Steven Bethard, Mona Diab, Mahmoud Ghoneim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, Alison Chang, and Pascale Fung. 2014. Overview for the first shared task on language identification in code-switched data. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, 62–72, Doha, Qatar. Association for Computational Linguistics.

Song, Ruihua, Haifeng Liu, Ji-Rong Wen, and Wei-Ying Ma. 2004. Learning block importance models for web pages. In *Proceedings of the 13th International Conference on the World Wide Web (WWW 2004)*, 203–211, New York, USA.

Souter, Clive, Gavin Churcher, Judith Hayes, Clive Souter, Gavin Churcher, Judith Hayes, John Hughes, and Stephen Johnson. 1994. Natural language identification using corpus-based models. *Hermes, Journal of Linguistics* 13.183–204.

Steinberger, Ralf, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufis, and Dániel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*, Geona, Italy.

Stupar, Marija, Tereza Jurić, and Nikola Ljubešić. 2011. Language identification of web data for building linguistic corpora. In *Proceedings of the 3rd International Conference on The Future of Information Sciences (INFuture 2011)*, 365–372, Zagreb, Croatia.

Sun, Fei, Dandan Song, and Lejian Liao. 2011. DOM based content extraction via text density. In *Proceedings of 34st International ACM-SIGIR Conference*

*on Research and Development in Information Retrieval (SIGIR'08)*, 245–254, Beijing, China.

Suzuki, Izumi, Yoshiki Mikami, Ario Ohsato, and Yoshihide Chubachi. 2002. A language and character set determination method based on $n$-gram statistics. *ACM Transactions on Asian Language Information Processing (TALIP)* 1.269–278.

Takçı, Hidayet, and Ekin Ekinci. 2012. Minimal feature set in language identification and finding suitable classification method with it. *Procedia Technology* 1.444–448.

——, and İbrahim Soğukpinar. 2004. Centroid-based language identification using letter feature set. In *Proceedings of the 5th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2004)*, 640–648, Seoul, Korea.

Takçı, Hidayet, and Tunga Güngör. 2012. A high performance centroid-based classification approach for language identification. *Pattern Recognition Letters* 33.2077–2084.

Tan, Liling, Marcos Zampieri, Nikola Ljubešić, and Jörg Tiedemann. 2014. Merging comparable data sources for the discrimination of similar languages: The dsl corpus collection. In *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)*, Reykjavik, Iceland.

Teahan, W. J. 2000. Text classification and segmentation using minimum cross-entropy. In *Proceedings of the 6th International Conference Recherche d' Information Assistee par Ordinateur (RIAO' 00)*, 943–961, College de France, France.

Teh, Yee Whye, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006a. Hierarchical Dirichlet processes. *Journal of the American Statistical Association* 101.1566–1581.

——, David Newman, and Max Welling. 2006b. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems 18 (NIPS-06)*, 1353–1360.

Tian, Jilei, and Janne Suontausta. 2003. Scalable neural network based language identification from written text. In *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, 48–51, Hong Kong.

Tiedemann, Jörg. 2009. *News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces*, volume V of *Amsterdam studies in the theory*

*and history of linguistic science*, 237–248. Amsterdam, The Netherlands: John Benjamins.

——. 2012. Parallel data , tools and interfaces in OPUS. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, 2214–2218, Istanbul, Turkey.

Tiedemann, Jörg, and Nikola Ljubešić. 2012. Efficient discrimination between closely related languages. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, 2619–2634, Mumbai, India.

Tran, Dat, and Dharmendra Sharma. 2005. Markov models for written language identification. In *Proceedings of the 12th International Conference on Neural Information Processing*, 67–70, Taipei, Taiwan.

Tran, Giang Binh, Dat Ba Nguyen, and Bin Thanh Kieu, 2010. *n*-gram based approach for multilingual language identification. Poster. available at `http://comp.mq.edu.au/programming/task_description/VILangTek.pdf`.

Trieschnigg, Dolf, Djoerd Hiemstra, Mariët Theune, Franciska Jong, and Theo Meder. 2010. An exploration of language identification techniques for the dutch folktale database. In *Proceedings of the LREC workshop Adaptation of Language Resources and Tools for Processing Cultural Heritage*, 2–6, Istanbul, Turkey.

Tromp, Erik, and Mykola Pechenizkiy. 2011. Graph-based *n*-gram language identification on short texts. In *Proceedings of Benelearn 2011*, 27–35, The Hague, Netherlands.

Tsoumakas, Grigorios, and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing & Mining* 3.1–13.

Tumasjan, Andranik, Timm O. Sprenger, Philipp G. Sandner, and Isabell M. Welpe. 2010. Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, 178–185, Washington DC, USA.

Tyers, Francis M, and Murat Serdar Alperen. 2010. South-east European times: A parallel corpus of Balkan languages. In *Proceedings of the LREC Workshop on Exploitation of Multilingual Resources and Tools for Central and (South-) Eastern European Languages*, 49–53, Valetta, Malta.

Ueda, N., and K. Saito. 2002. Parametric mixture models for multi-labeled text. In *Advances in Neural Information Processing Systems 14 (NIPS-02)*, 721–728, Vancouver, Canada.

Ueda, Yoshido, and Seiichi Nakagawa. 1990. Prediction for phoneme/syllable/word-category and identification of language using HMM. In *Proceedings of the 1990 International Conference on Spoken Language Processing, volume 2*, 1209–1212, Kobe, Japan.

van Noord, Gertjan, 1994. *TextCat*. Software available at `http://software.wise-guys.nl/libtextcat/`.

van Rijsbergen, C. J. 1979. *Information Retrieval*. London, UK: Butterworths.

Vatanen, Tommi, Jaakko J. Vayrynen, and Sami Virpioja. 2010. Language identification of short text segments with $n$-gram models. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, 3423–3430, Valetta, Malta.

Vogel, John, and David Tresner-Kirsch. 2012. Robust language identification in short, noisy texts: Improvements to LIGA. In *Proceedings of the 3rd International Workshop on Mining Ubiquitous and Social Environments (MUSE)*, 1–9, Bristol, UK.

Vojtek, Peter, and Mária Bieliková. 2007. Comparing natural language identification methods based on Markov processes. In *Proceedings of the Fourth International Seminar on Computer Treatment of Slavic and East European Languages (SLOVKO 2007)*, 126–138, Bratislava, Slovakia.

Weninger, Tim, William H Hsu, and Jiawei Han. 2010. CETR: content extraction via tag ratios. In *Proceedings of the 19th International Conference on the World Wide Web (WWW 2010)*, 971–980, Raleigh, USA.

Windisch, Gergely, and László Csink. 2005. Language identification using global statistics of natural languages. In *Proceedings of the 2nd Romanian-Hungarian Joint Symposium on Applied Computational Intelligence (SACI)*, 243–255, Timisoara, Romania.

Winkelmolen, Fela, and Viviana Mascardi. 2011. Statistical language identification of short texts. In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence*, 498–503, Rome, Italy.

Winnemöller, Ronald. 2010. Drive-by language identification. In *Proceedings of the 11th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2010)*, 494–502, Iasi, Romania.

Witten, Ian H, Alistair Moffat, and Timothy Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. San Francisco, USA: Morgan Kaufmann.

Wolpert, David H. 1992. Stacked generalization. *Neural Networks* 5.241–259.

Xafopoulos, Alexandros, Constantine Kotropoulos, George Almpanidis, and Ioannis Pitas. 2004. Language identification in web documents using discrete HMMs. *Pattern Recognition* 37.583–594.

Xia, Fei, Carrie Lewis, and William D. Lewis. 2010a. Language ID for a thousand languages. In *LSA Annual Meeting Extended Abstracts*, 1–4.

——, Carrie Lewis, and William D Lewis. 2010b. The problems of language identification within hugely multilingual data sets. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, 2790–2797, Valetta, Malta.

——, William Lewis, and Hoifung Poon. 2009. Language ID in the context of harvesting language data off the web. In *Proceedings of the 12th Conference of the EACL (EACL 2009)*, 870–878, Athens, Greece.

Yaghan, Mohammad Ali. 2008. "arabizi" : A contemporary style of Arabic slang. *Design Issues* 24.39–52.

Yamaguchi, Hiroshi, and Kumiko Tanaka-Ishii. 2012. Text segmentation by language using minimum description length. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, 969–978, Jeju Island, Korea.

Yang, Xi, and Wenxin Liang. 2010. An n-gram-and-wikipedia joint approach to natural language identification. In *Proceedings of the 4th International Universal Communication Symposium (IUCS 2010)*, 332–339, Beijing, China.

Yang, Yiming, and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, 412–420, Nashville, USA.

Yeh, Alexander. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, 947–953, Saarbrücken, Germany.

Zaidan, Omar F., and Chris Callison-Burch. 2011. The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 37–41, Portland, Oregon, USA. Association for Computational Linguistics.

Zaidan, Omar F, and Chris Callison-Burch. 2014. Arabic dialect identification. *Computational Linguistics* 40.171–202.

Zampieri, Marcos. 2013. Using bag-of-words to distinguish similar languages: How efficient are they? In *Proceedings of the 2013 IEEE 14th International Symposium on Computational Intelligence and Informatics (CINTI)*, 37–41, Budapest, Hungary.

——, Binyam Gebrekidan Gebre, and Sascha Diwersy. 2012a. Classifying pluricentric languages: Extending the monolingual model. In *Proceedings of the Fourth Swedish Language Technlogy Conference (SLTC2012)*, 79–80, Lund, Sweden.

——, ——, and ——. 2013. N-gram language models and POS distribution for the identification of Spanish varieties. In *Proceedings of la 20ème conférence du Traitement Automatique du Langage Naturel (TALN 2013)*, 580–587, Sable d'Olonne, France.

——, Binyam Gebrekidan Gebre, and Holland Nijmegen. 2012b. Automatic identification of language varieties: The case of Portuguese. In *Proceedings of The 11th Conference on Natural Language Processing (KONVENS 2012)*, 233–237, Vienna, Austria.

——, Liling Tan, Nikola Ljubešić, and Jörg Tiedemann. to appear. A report on the DSL shared task 2014. In *Proceedings of the 1st Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, Dublin, Ireland.

——, Liling Tan, Nikola Ljubešić, and Jörg Tiedemann. 2014. A report on the dsl shared task 2014. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, 58–67, Dublin, Ireland.

Zubiaga, Arkaitz, Inaki San Vicente, Pablo Gamallo, José Ramom Pichel, Inaki Alegria, Nora Aranberri, Aitzol Ezeiza, and Victor Fresno. 2014. Overview of TweetLID: Tweet language identification at SEPLN 2014. In *Proceedings of the Tweet Language Identification Workshop co-located with 30th Conference of the Spanish Society for Natural Language Processing (SEPLN 2014)*, 1–11, Girona,Spain.